TRƯỜNG ĐẠI HỌC CHU VĂN AN
KHOA CÔNG NGHỆ THÔNG TIN

Ts. DƯƠNG XUÂN THÀNH

GIÁO TRÌNH
TIN HỌC ĐẠI CƯƠNG

(Soạn theo chương trình đã được Bộ GD&ĐT phê chuẩn)
In lần thứ ba có sửa chữa bổ sung

NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT

TRƯỜNG ĐẠI HỌC CHU VĂN AN
KHOA CÔNG NGHỆ THÔNG TIN

Ts. DƯƠNG XUÂN THÀNH

Giáo trình
TIN HỌC ĐẠI CƯƠNG

(Soạn theo chương trình đã được Bộ GD&ĐT phê chuẩn)

In lần thứ ba – có sửa chữa, bổ sung

NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT
HÀ NỘI

Lời nói đầu cho lần xuất bản thứ ba

Giáo trình Tin học đại cương xuất bản lần thứ hai đến nay đã qua 4 năm, một số nội dung đã trở nên lạc hậu. Trong bốn năm qua tác giả đã nhận được nhiều ý kiến đóng góp quý báu về nội dung và hình thức cuốn sách. Để chuẩn bị cho lần xuất bản thứ 3, tác giả đã biên soạn lại nhiều nội dung, đặc biệt đã đưa vào phần giới thiệu về Hệ điều hành Windows. Phần Ngôn ngữ lập trình Pascal cũng được bổ sung một số nội dung và bài thực hành.

Bố cục của giáo trình được soạn theo đúng nội dung môn học đã được Bộ Giáo dục và Đào tạo phê duyệt đầu năm 2008.

Giáo trình được xuất bản lần này có sự hợp tác chặt chẽ giữa Tác giả và Ban lãnh đạo Trường Đại học Chu Văn An nhằm bổ sung một giáo trình hoàn chỉnh vào tủ sách của Trường. Tác giả xin chân thành cảm ơn Tiến sĩ Đặng Văn Định, Chủ tịch Hội đồng quản trị, Tiến sĩ Tạ Tiến Hùng, Hiệu trưởng Trường Đại học Chu Văn An và các thành viên khác đã giúp đỡ việc in ấn cuốn sách.

Mục đích của lần xuất bản này không phải chỉ nhằm cung cấp một giáo trình cho sinh viên Đại học Chu Văn An mà còn nhằm vào đối tượng là học sinh phổ thông, sinh viên đại học và những người lần đầu tiên làm quen với môn Tin học. Các ví dụ trong giáo trình chủ yếu nhằm minh hoạ lý thuyết.

Giáo trình cũng có thể sử dụng như là tài liệu tham khảo trong công tác giảng dạy môn "Nhập môn Tin học" và "Ngôn ngữ lập trình Pascal" cho giáo viên Tin học tại các Trung tâm và các Trường.

Nội dung giáo trình bao gồm ba phần cơ bản:

- Các kiến thức toán học hỗ trợ
- Các kiến thức về lập trình
- Các kiến thức phục vụ trực tiếp công việc soạn thảo văn bản.

Tác giả hy vọng giáo trình sẽ giúp ích cho bạn đọc trong quá trình học tập môn Tin học, đặc biệt là với những người lần đầu tiên làm quen với môn học này.

Mọi ý kiến đóng góp xin gửi về địa chỉ:

Khoa Công nghệ Thông tin, Trường Đại học Chu Văn An

2A - Đường Bạch Đằng – Thị xã Hưng Yên

Hoặc điện thoại: ***04 – 8767307***

Tác giả xin trân trọng cảm ơn.

Hưng Yên, tháng 3 năm 2008

MỤC LỤC

PHẦN 1
ĐẠI CƯƠNG VỀ TIN HỌC VÀ MÁY VI TÍNH

Bài mở đầu
THÔNG TIN VÀ TIN HỌC

I. LỊCH SỬ PHÁT TRIỂN CỦA TIN HỌC VÀ MÁY VI TÍNH

Toán học nghiên cứu các con số, Hoá học nghiên cứu các chất... vậy Tin học nghiên cứu cái gì? Nước Pháp là nơi đầu tiên đặt tên gọi cho ngành Tin học, người ta định nghĩa "***Tin học là tập hợp các khoa học, kỹ thuật về việc xử lý tự động và hợp lý thông tin mang kiến thức của con người để lưu giữ và truyền gửi chúng***".

Từ điển Bách khoa 1986 của Liên Xô (cũ) định nghĩa :" ***Tin học là một lĩnh vực khoa học nghiên cứu cấu trúc và tính chất của thông tin khoa học, cùng với việc thu thập, xử lý, lưu giữ, biến đổi và truyền chúng***". Tuy định nghĩa có thể khác nhau nhưng nội dung của các định nghĩa ấy đều thống nhất ở một điểm: Tin học là khoa học chuyên nghiên cứu về một đối tượng cụ thể: đó là thông tin và dữ liệu. Công cụ không thể thiếu đối với người nghiên cứu Tin học là máy tính và các ngôn ngữ giao tiếp. Máy tính bao gồm hai phần: phần cứng và phần mềm. Phần cứng gồm một bộ vi xử lý, các mạch kết nối, các thiết bị ngoại vi, phần mềm là các chương trình được lập ra bằng một ngôn ngữ nào đó, được ghi vào đĩa mềm hoặc đĩa cứng và được gọi ra theo yêu cầu của người sử dụng.

Từ thời xa xưa con người đã biết dùng các công cụ tính toán như que tính, bàn tính ... Các máy tính mà chúng ta đang sử dụng mới chỉ có lịch sử phát triển khoảng 50 năm trở lại đây song nó đã mang lại một cuộc cách mạng thực sự trong đời sống xã hội loài người, điều mà chưa một ngành khoa học nào khác có thể đạt được trong một khoảng thời gian ngắn như vậy. Trong gần 50 năm qua máy vi tính đã trải qua tới 5 thế hệ.

* 1950-1958, thế hệ thứ nhất. Các máy tính thế hệ này được lắp ráp bằng đèn điện tử chân không, số liệu được đưa vào bằng bìa đục lỗ, tốc độ tính vào khoảng 300-3000 phép/s.

* 1958-1964, thế hệ thứ hai. Bộ xử lý trung tâm được lắp ráp bằng mạch bán dẫn, số liệu được đưa vào bằng bìa và băng đục lỗ. Bộ nhớ ngoài bằng trống từ

hoặc băng từ. Tốc độ tính toán vào khoảng 10 000 - 100 000 phép/s. Máy đã có chương trình dịch và hệ điều hành đơn giản.

* 1965-1974, thế hệ thứ ba. Bộ xử lý lắp bằng vi mạch điện tử cỡ nhỏ, bộ nhớ trong bằng xuyến từ hoặc màng mỏng từ, bộ nhớ ngoài bằng đĩa cứng, tốc độ tính đạt từ 100 000 đến 1 triệu phép/s. Máy đã có hệ điều hành đa chương trình, có các kênh vào ra để điều khiển máy in và bộ nhớ ngoài.

* 1974 đến nay, thế hệ thứ tư và khởi đầu của thế hệ thứ năm. Các máy tính thế hệ này sử dụng các bộ vi xử lý được lắp ráp bằng các vi mạch nhỏ, các vi mạch này được chế tạo trong công nghệ vi mạch cỡ lớn và rất lớn. Bộ nhớ ngoài dùng đĩa mềm và đĩa cứng. Máy thường được chế thành hai loại: loại dùng thông thường cho một người, đa chức năng (Personal Computer) và loại chuyên dụng đa chương trình, đa vi xử lý.

Máy tính thế hệ thứ năm sẽ là các máy tính có trí khôn nhân tạo, có những hệ suy diễn phát triển và những hệ quản lý cơ sở kiến thức. Người ta hy vọng rằng máy tính thế hệ này sẽ biết suy diễn để chọn tìm kiến thức và tự định ra các thuật toán giải quyết các bài toán thực tế. Đã xuất hiện một số máy vi tính không dùng bàn phím, người sử dụng ra lệnh cho máy bằng tiếng nói, nếu máy chưa hiểu đúng mệnh lệnh nó sẽ yêu cầu nhắc lại . Tuy nhiên những máy loại này cho đến nay vẫn chưa có khả năng hiểu được toàn bộ ngôn ngữ của con người.

Bước sang thế kỷ 21 sự phát triển của Tin học ngày càng nhanh hơn, phong phú hơn và cũng chứa đựng nhiều nguy cơ hơn. Mạng Internet đã trở thành công cụ không thể thiếu với các quốc gia muốn phát triển khoa học, giáo dục, kinh tế, thương mại. Internet đã trở thành phương tiện giao lưu văn hoá, tình cảm của con người không phân biệt người đó đang sống ở nơi nào trên thế giới. Mặt trái của quá trình Tin học hoá là con người lệ thuộc quá nhiều vào kỹ thuật, vào máy móc, sự cố trên một máy chủ có thể làm ngừng trệ các giao dịch thương mại, các quá trình sản xuất, truyền dẫn thông tin và hậu quả không thể tính hết bằng tiền. Xã hội hiện đại xuất hiện loại tội phạm mới đó là tội phạm công nghệ cao, những kẻ phạm tội đều là những kẻ được trang bị các kiến thức Tin học hoàn chỉnh. Học tập để nắm bắt được các kiến thức hiện đại nhưng cũng cần học tập để điều chỉnh hành vi khi làm chủ được công nghệ.

II. TIN - BẢN TIN

Tin là khái niệm dùng để mô tả những hiện tượng, sự vật hoặc hiểu biết, nhận thức của con người.

Bản tin là một tập hợp các **Tin** dưới dạng mã hoá nhằm mô tả một quá trình diễn biến nào đó, ví dụ: một bản vẽ, một đoạn chương trình, một văn bản .. Con người trao đổi thông tin với nhau bằng các phương tiện mã hoá như chữ viết, tín hiệu ánh sáng, hình vẽ... Sự giao tiếp giữa con người và các thiết bị tin học chủ yếu thông qua phương pháp đối thoại. Người sử dụng (NSD) ra các lệnh cần thiết thông qua phương tiện là các ngôn ngữ đối thoại (ngôn ngữ này có thể là các

tiếng nói, chữ viết hoặc biểu tượng đồ hoạ), máy thực hiện các lệnh đó và cho ta các kết quả xử lý. Hiện thời bộ não của máy tính chưa đạt đến độ hoàn thiện như bộ não con người do vậy con người không thể ra lệnh cho máy tính bằng ngôn ngữ thông thường. Các thiết bị điện dù hiện đại đến máy cũng chỉ có thể làm việc khi có nguồn điện cung cấp, ngắt điện máy tính trở thành một vật vô tri. Vì lẽ đó con người phải nghĩ ra một ngôn ngữ gọi là ngôn ngữ máy, ngôn ngữ này dựa trên hai ký tự là 0 và 1, một cách hình thức chúng ta coi ký tự 0 ứng với trạng thái không có điện, còn ký tự 1 ứng với trạng thái có điện. Để tiện thay vì gọi ký tự 0, ký tự 1 chúng ta sẽ gọi là số 0, số 1. Bên trong máy tính các lệnh hoặc dữ liệu được biểu diễn dưới dạng các số 0 và 1 này. Cũng giống như hệ đếm thập phân, hệ đếm La Mã, hai số 0 và 1 tạo nên hệ đếm cơ số 2 mà ta quen gọi là hệ đếm nhị phân, hai chữ số này tương ứng với hai trạng thái ngắt - đóng mạch điện, điều này ta sẽ nghiên cứu kỹ hơn ở phần sau.

Chương 1
CƠ SỞ TOÁN CỦA TIN HỌC

I. KHÁI NIỆM HỆ ĐẾM

Hệ đếm được hiểu là một tập hợp các ký hiệu và những quy tắc dùng để biểu diễn và tính toán giá trị các số. Chúng ta đã quen với hai hệ đếm thông dụng là hệ đếm thập phân và hệ đếm La Mã.

Hệ đếm La Mã dùng năm ký hiệu mà ta sẽ gọi là năm số cơ sở :

I = 1; V = 5; X = 10; D = 100; M = 1000

Giá trị của các số I, V,... không phụ thuộc vào vị trí của nó trong chữ số như các số trong hệ đếm cơ số 10. Các số khác của hệ đếm La Mã được hình thành bằng cách ghép các số cơ sở theo nguyên tắc :

* Các số giống nhau đặt liền nhau tạo nên một chữ số có giá trị bằng tổng của các chữ số cơ sở, ví dụ : III = 1+1+1 = 3; XXX = 10+10+10 = 30.

* Hai số khác nhau đứng liền nhau sẽ cho ta giá trị bằng tổng của chúng nếu số lớn đứng trước, số bé đứng sau; cho giá trị hiệu nếu số lớn đứng sau số bé đứng trước, ví dụ : IX = 10-1 = 9; XI = 10+1 = 11.

Hệ đếm thập phân dùng 10 số cơ sở 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Giá trị của mỗi số cơ sở phụ thuộc vào vị trí của nó trong chữ số. Nếu các số trong chữ số là giống nhau thì giá trị của số đứng bên trái sẽ lớn gấp 10 số đứng bên phải, ví dụ : 999, số 9 đầu tiên bên phải có giá trị bằng 9 đơn vị, số 9 thứ hai có giá trị bằng 90 đơn vị và số 9 thứ 3 có giá trị bằng 900 đơn vị.

Trong hệ đếm thập phân các con số được tạo ra bằng cách ghép quay vòng các chữ số cơ sở bắt đầu từ chữ số thứ 2 tức là chữ số 1, ví dụ:

* Vòng ghép thứ nhất:

ghép số thứ hai (tức là số 1) với các số cơ sở ta được: 10,11,12...19

sau đó ghép chữ số thứ ba (số 2) ta sẽ có: 20,21,22...29

Loạt số cuối trong vòng vòng ghép đầu tiên là 90,91,92...99

* Vòng ghép thứ hai:

Trong vòng ghép thứ hai ta sẽ lấy lần lượt các số của vòng ghép thứ nhất tức là số 10,11,.., 99 ghép với các chữ số cơ sở để tạo ra các chữ số mới:

100,101,...109

110,111,...119

.....

990,991....999.

Tóm lại nguyên tắc ghép là: lấy các số tạo ra trong vòng ghép trước ghép với các số cơ sở để tạo ra các số tiếp theo.

Với một số trong hệ thập phân ví dụ 276,52 ta quy định vị trí của các chữ số như sau: Chữ số bên trái sát dấu phảy thập phân có vị trí 0 (số 6) tiếp đó là vị trí 1 (số 7) vị trí 2 (số 2) .... các số bên phải dấu phảy sẽ có vị trí âm. Ta có thể biểu diễn số 276,52 như sau;

$$276,52 = 200 + 70 + 6 + 0,5 + 0,02 = 2.10^2 + 7.10^1 + 6.10^0 + 5.10^{-1} + 2.10^{-2}$$

Nếu chú ý một chút ta sẽ thấy các số 10^2 , 10^1 , 10^0 , 0^{-1}, 10^{-2} chính là luỹ thừa của cơ số hệ đếm (cơ số ở đây là 10) , số mũ của luỹ thừa là số chỉ vị trí của bản thân từng chữ số .

Tổng quát với một hệ đếm có cơ số k (k > 1) ta có thể biểu diễn số:

$N = (a_n a_{n-1} a_{n-2} \ldots a_0 a_{-1} a_{-2} \ldots a_{-m})_k$ theo công thức:

$$N = a_n.k^n + a_{n-1}.k^{n-1} + ... + a_0.k^0 + a_{-1}.k^{-1} + ... + a_{-m}.k^m \qquad (1.1)$$

II. HỆ ĐẾM NHỊ PHÂN - BIT, BYTE, WORD

Các máy tính không biết đếm như con người. Máy tính làm việc khi có điện và ngừng làm việc khi mất điện, nghĩa là chỉ có hai trạng thái Đóng-Ngắt. Các vi mạch hoạt động cũng chỉ có hai trạng thái: có dòng điện chạy qua và không có dòng điện chạy qua. Vì lẽ đó để máy có thể đếm, cần tạo ra một hệ đếm chỉ có hai chữ số tức là hệ đếm nhị phân. Hệ đếm nhị phân dùng trong máy vi tính hiện nay gồm hai chữ số 0 và 1. Để tiện cho các diễn giải sau này ta sẽ viết các chữ số kèm theo một chỉ số với các hệ đếm không phải là thập phân, cụ thể: dùng chỉ số 16 cho hệ đếm cơ số 16 (tức là số học HEX - Hexadecimal) , chỉ số 2 cho hệ đếm nhị phân (Binary) và chỉ số 8 cho hệ bát phân. Ví dụ: 1_{16} là số 1 trong hệ đếm cơ số 16, 1_2 là số 1 trong hệ nhị phân, với hệ đếm thập phân có thể dùng chỉ số 10 (Decimal) hoặc không dùng, ví dụ 1 và 1_{10} là số 1 trong hệ thập phân .

Việc tạo ra các chữ số mới trong hệ nhị phân cũng áp dụng nguyên tắc ghép như với hệ đếm cơ số 10, cụ thể:

Chữ số cơ sở: 0, 1

Vòng ghép thứ 1: 10, 11

Vòng ghép thứ 2: 100, 101, 110, 111

Vòng ghép thứ 3: 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111

.....

Nếu ta xem số 0 và số 1 của hệ nhị phân là bằng số 0 và số 1 của hệ thập phân thì có thể thấy 16 số đầu của hệ thập phân sẽ tương ứng bằng các số nhị phân viết trong bảng 1.1:

Bảng 1.1

Hệ 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hệ 2	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111

Đơn vị đo dung lượng thông tin được định nghĩa là bit. Một bit được hiểu là dung lượng cần thiết để lưu trữ được một chữ số nhị phân 0 hoặc 1, như vậy để biểu diễn 10 chữ số thập phân đầu tiên (0-9) ta cần chữ số nhị phân có độ dài 4 bit (số $9_{10} = 1001_2$). Trong các máy vi tính hiện nay để viết các số có độ dài lớn hơn người ta thường dùng bội số của 4 ví dụ 8 bit, 16 bit, 32 bit, 64 bit.

Số không dấu nhỏ nhất trong các số nhị phân 8 bit là 0000 0000 và số lớn nhất 1111 1111, các số này trong hệ thập phân sẽ là 0 và 255.

Để biểu diễn những đơn vị dung lượng lớn hơn người ta dùng các bội số của Bit

1 byte (đọc là bai) $=2^3$ bits = 8 bits

1 word = 2^4 bits = 2 bytes = 16 bits

1 KB = 2^{10} bytes = 1024 bytes

1 MB = 2^{10} KB = 1024*1024 = 1 048 576 bytes

1 GB = 2^{10} MB = 1024 MB = 1 073 741 824 bytes

III. HỆ ĐẾM CƠ SỐ 8

Hệ đếm cơ số 8 là hệ đếm hình thành từ 8 chữ số cơ sở : 0 1 2 3 4 5 6 7, theo nguyên tắc ghép quay vòng dễ dàng tạo ra các chữ số tiếp theo của hệ đếm này, ví dụ sau số 7 sẽ là các số :

10 11 12 13 14 15 16 17 (xem bảng 1.2)

Bảng 1.2

Hệ 2	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111
Hệ 8	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
Hệ 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hệ 16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

IV. HỆ ĐẾM CƠ SỐ 16

Hệ đếm cơ số 16 là hệ đếm gồm 16 chữ số cơ sở, chúng ta sẽ không đề cập đến vấn đề tại sao lại phải dùng hệ đếm 16 mà chỉ nghiên cứu sơ bộ việc đổi các số từ hệ 16 sang các hệ khác và ngược lại.

Hệ đếm cơ số 16 (Hexadecimal) bao gồm các chữ số sau:

0 1 2 3 4 5 6 7 8 9 A B C D E F

Theo phương pháp ghép quay vòng đã nêu ta sẽ có:

$10_{16} = 16_{10}$, $11_{16} = 17_{10}$, $12_{16} = 18_{10}$, $13_{16} = 19_{10}$, $14_{16} = 20_{10}$.....

V. ĐỔI SỐ GIỮA CÁC HỆ ĐẾM

1. Đổi số giữa hệ 2 và hệ 10

a. Đổi từ hệ 2 sang hệ 10

Trong hệ đếm nhị phân cơ số là 2, áp dụng công thức (1.1) ta có thể đổi số nhị phân thành số thập phân, ví dụ đổi số 1100_2

$1100_2 = 1.2^3 + 1.2^2 + 0.2^1 + 0.2^0$

$= 8 + 4 + 0 + 0 = 12_{10}$

Vậy 1100 trong hệ nhị phân chính là 12 trong hệ thập phân.

Xét thêm ví dụ với các số lớn nhất trong các số nhị phân 8 bit và 16 bit tức là các số 1111 1111 và 1111 1111 1111 1111:

$1111\ 1111_2 = 1\times2^7 + 1\times2^6 + 1\times2^5 + 1\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0$

$= 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255_{10}$

$1111111111111111_2 = 1\times2^{15} + 1\times2^{14} + ... + 1\times2^1 + 1\times2^0 =$

$= 32768 + 16384 + 8192 + ... + 0 = 65535_{10}$

**** Đổi các số nhị phân lẻ***: Với những số nhị phân có phần lẻ cách đổi cũng hoàn toàn tương tự, ví dụ muốn đổi số 1011.01b ta viết:

$1011.01_2 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 + 0\times2^{-1} + 1\times2^{-2}$

$= 8 + 0 + 2 + 1 + 0 + 0.25_{10}$

$1011.01_2 = 11.25_{10}$

Bằng phương pháp tương tự ta có thể kiểm tra sự chính xác của các số liệu đã ghi trong bảng 1.

b. Đổi số từ hệ 10 sang hệ 2

Để đổi một số từ hệ thập phân sang hệ nhị phân ta dùng phương pháp chia liên tiếp số đó và các kết quả của phép chia cho 2 chừng nào kết quả chia vẫn còn khác 0. Ghép các số dư theo chiều ngược lại ta được số nhị phân chuyển đổi. Ví dụ đổi số 156 sang hệ nhị phân:

156 : 2 = 78 dư 0
78 : 2 = 39 dư 0
39 : 2 = 19 dư 1
19 : 2 = 9 dư 1
9 : 2 = 4 dư 1
4 : 2 = 2 dư 0
2 : 2 = 1 dư 0
1 : 2 = 0 dư 1

Vậy $156_{10} = 1001\ 1100_2$

Với những số thập phân ***có phần lẻ*** việc chuyển đổi phần lẻ không phải là chia mà là nhân liên tiếp phần lẻ với 2 và giữ lại phần nguyên của các kết quả nhân. Quá trình nhân kết thúc đến khi đạt được một trong hai khả năng dưới đây :

a. Phần lẻ của phép nhân có giá trị bằng 0, ví dụ đổi số 0.67875 sang hệ nhị phân

0.67875 x 2 = 1.375 phần nguyên = 1
0.3750 x 2 = 0.75 phần nguyên = 0
0.75 x 2 = 1.5 phần nguyên = 1
0.5 x 2 = 1.0 phần nguyên = 1

phần lẻ bằng 0 vậy $0.67875_{10} = 0.1011_2$

b. Số chữ số nhị phân tìm được đã đạt độ chính xác cần thiết hoặc đã bằng số chữ số nhị phân mà máy có thể xử lý. Ví dụ máy của chúng ta chỉ xử lý được 8 bit, ta cần đổi số 0.68 sang hệ 2.

0.68 × 2 = 1.36 phần nguyên = 1
0.36 × 2 = 0.72 phần nguyên = 0
0.72 × 2 = 1.44 phần nguyên = 1
0.44 × 2 = 0.88 phần nguyên = 0
0.88 × 2 = 1.76 phần nguyên = 1
0.76 × 2 = 1.52 phần nguyên = 1

$0.52 \times 2 = 1.04$ phần nguyên = 1

$0.04 \times 2 = 0.08$ phần nguyên = 0

Vậy $0.68_{10} = 0.1010\ 1110_2$

2. Đổi số giữa hệ 8 và hệ 10

Xét số 17 trong hệ 8, số 7 có vị trí 0 và số 1 có vị trí 1 do đó số 17 chuyển sang hệ 10 sẽ là :

$17_8 = 1*8^1 + 7*8^0 = 8 + 7 = 15_{10}$

Đổi số 152 từ hệ 8 sang hệ 10

$152_8 = 1*8^2 + 5*8^1 + 2*8^0 = 64 + 40 + 2 = 106_{10}$

Việc đổi số ngược lại từ hệ 10 sang hệ 8 cũng tuân theo nguyên tắc chia như đã nêu trong mục IV-1-b . Xét ví dụ đổi số 106 từ hệ 10 sang hệ 8

106 : 8 = 13 dư 2

13: 8 = 1 dư 5

1:8 = 0 dư 1

Ghép các số dư theo chiều từ dưới lên ta có $106_{10} = 152_8$

3. Đổi số giữa hệ 16 và hệ 10

Xét số $1F_{16}$, theo phương pháp đã nêu trên số 1 có vị trí 1 số F có vị trí 0, áp dụng nguyên tắc đổi đã biết ta có:

$$1F_{16} = 1.16^1 + F.16^0 = 16 + 15 = 31_{10}$$

Vậy $1F_{16} = 31_{10}$

Việc đổi số từ hệ 10 sang hệ 16 cũng áp dụng nguyên tắc chia số của hệ đếm 10 cho cơ số của hệ 16 tức là chia cho 16, ví dụ:

31 : 16 = 1 dư 15

1 : 16 = 0 dư 1

Vì số 15 của hệ 16 là bằng F nên ghép các số theo chiều từ dưới lên ta có:

$$31_{10} = 1F_{16}$$

4. Đổi số giữa hệ 8 và hệ 2

Tám chữ số cơ sở của hệ đếm cơ số 8 chuyển sang hệ 2 chỉ dùng nhiều nhất là 3 bit (xem bảng 2) do vậy khi đổi một số từ hệ 8 sang hệ 2 ta đổi từng chữ số sang dạng mã 3 bit , sau đó ghép kết quả lại theo thứ tự.

Ví dụ đổi số 23_8 sang hệ 2. Số 2_8 đổi sang hệ 2 dưới dạng mã 3 bit là 010, số 3_8 là 011.

Vậy số $23_8 = 010\ 011_2$

Có thể kiểm tra kết quả bằng cách đổi tất cả sang hệ 10:

$$23_8 = 2*8^1 + 3*8^0 = 19_{10}$$

$$010011_2 = 1*2^4 + 1*2^1 + 1*2^0 = 16+2+1 = 19_{10}$$

Với các số hệ 2 khi đổi sang hệ 8 ta phân thành từng nhóm 3 bit tính từ phải qua trái, nhóm cuối cùng có thể nhỏ hơn 3 bit, đổi từng nhóm 3 bit sang hệ 8 bằng cách tra cứu trong bảng 2 sau đó ghép các kết quả lại.

5. Đổi số giữa hệ 16 và hệ 2

Để biểu diễn một trong 16 chữ số đầu tiên của hệ 16 sang hệ 2 ta cần số nhị phân có độ dài 4 bit. Với các số hệ 16 có 1 chữ số (tức là với 16 chữ số cơ sở) ta có thể dùng ngay bảng 2 để đổi.

Với các số có từ hai chữ số trở lên ta áp dụng phương pháp đổi từng chữ số sang mã 4 bit, ví dụ đổi số AF_{16} sang hệ nhị phân. Tra bảng 2 ta có :

$$A = 1010;\quad F = 1111$$

Vậy $AF_{16} = 1010\ 1111_2$

Có thể dễ dàng kiểm tra lại kết quả đã có bằng cách chuyển tất cả sang hệ thập phân:

$$AF_{16} = A.16^1 + F.16^0 = 10.16 + 15.1 = 175_{10}$$

$$1010\ 1111_2 = 2^7 + 2^5 + 2^3 + 2^2 + 2^1 + 2^0 = 175_{10}$$

Để đổi một số từ hệ 2 sang hệ 16 ta áp dụng nguyên lý sau đây:

* Nếu số hệ 2 viết dưới dạng mã nhỏ hơn 4 bit thì tra bảng 2 để đổi.

* Nếu số hệ 2 viết dưới dạng mã lớn hơn 4 bit thì phân chữ số hệ 2 thành từng cụm 4 bit ***tính từ phải qua trái***, cụm cuối cùng bên trái có thể có số bit nhỏ hơn 4. Mỗi cụm này ứng với một chữ số của hệ 16 (xem bảng 2). Ghép lần lượt các chữ số của hệ 16 lại ta được kết quả chuyển đổi.

Ví dụ đổi số : 10 1110 1101

Chia số trên thành 3 cụm là 1101, 1110, 10, tra bảng 2 ta có:

$$1101 = D,\quad 1110 = E \quad \text{và } 10 = 2$$

Vậy $10\ 1110\ 1101_2 = 2ED_{16}$

6. Đổi số giữa hệ 8 và hệ 16

Như đã trình bày các chữ số cơ sở của hệ 8 khi chuyển sang hệ 2 sẽ được biểu diễn dưới dạng mã nhị phân 3 bit, còn các chữ số cơ sở của hệ 16 được biểu diễn dưới dạng mã 4 bit (xem bảng 1.3). Việc chuyển đổi các số giữa hệ 8 và hệ 16 sẽ tiến hành thông qua hệ 2 theo phương pháp sau đây:

Bảng 1.3

Hệ 2	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111
Hệ 8	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
Hệ 16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

a. Đổi hệ 8 sang hệ 16

* Đổi từng chữ số của hệ 8 sang hệ 2 (theo bảng 1.3)

* Ghép các số nhị phân theo thứ tự

* Chia số nhị phân vừa tạo ra thành từng nhóm 4 bit tính từ phải qua trái

* Dùng bảng 3 đổi từng nhóm 4 bit này thành số hệ 16, sau đó ghép kết quả lại theo thứ tự

Ví dụ : đổi 35_8 sang hệ 16

Tra bảng 3 ta có $3_8 = 011_2$, 5_8 101_2 do đó:

$$35_8 = 011101_2 = 0001\ 1101_2$$

Tra tiếp bảng 3 ta có $0001_2 = 1_{16}$, $1101_2 = D_{16}$

Vậy $35_8 = 1D_{16}$

Bạn đọc có thể kiểm tra lại kết quả bằng cách chuyển đổi tất cả sang hệ 10.

b. Đổi số hệ 16 sang hệ 8

Quá trình đổi số hệ 16 sang hệ 8 là quá trình ngược lại với những điều đã trình bày trong mục a, do vậy ở đây chỉ trình bày một ví dụ minh hoạ.

Ví dụ đổi số $A2B_{16}$ sang hệ 8

Tra bảng 3 ta có : $A_{16} = 1010$, $2_{16} = 0010$, $B_{16} = 1011$ do đó:

$$A2B_{16} = 1010\ 0010\ 1011_2$$

Chia số 1010 0010 1011 thành từng nhóm 3 bit ta có:

000 101 011

Tra bảng 3 : $101_2 = 5_8$, $000_2 = 0_8$, $011_2 = 3_8$

Vậy : $A2B_{16} = 5053_8$

Thử lại : $A2B_{16} = A*16^2 + 2*16 + B = 2560+32+11 = 2603_{10}$

$5053_8 = 5*8^3 + 0*8^2 + 5*8 + 3 = 2560+40+3 = 2603_{10}$

Lưu ý :

Ngoài cách nêu trên còn có thể dùng phương pháp chuyển các số đã cho sang hệ 10 sau đó từ hệ 10 chuyển tiếp sang hệ đếm mới, ví dụ đổi số $A2B_{16}$ sang hệ 8:

$$A2B_{16} = A.16^2 + 2.16^1 + B.16^0 = 10.256 + 2.16 + 11.1 = 2603_{10}$$

Chia liên tiếp 2603 cho 8 ta được:

2603 : 8 = 325 dư 3

325 : 8 = 40 dư 5

40 : 8 = 5 dư 0

5 : 8 = 0 dư 5

Ghép các số từ dưới lên ta được $A2B_{16} = 2603_{10} = 5035_8$

7. Biểu diễn số âm trong hệ nhị phân

Các số nhị phân đã xét đều là các số không dấu tức là các số không âm. Để biểu diễn các số âm chúng ta phải quy định bit mang dấu. Trong phạm vi giáo trình này chúng ta chỉ xét một số mã nhị phân thông dụng đó là các mã 4 bit, 8 bit, 16 bit.

Trong các mã nêu trên nếu quy định vị trí của **bit đầu tiên bên phải** là vị trí 0 thì vị trí của các **bit bên trái nhất** sẽ lần lượt là 3, 7, 15.

Bit bên trái nhất của mỗi số được quy định là bit mang dấu, **bit này sẽ mang dấu cộng nếu nó có giá trị bằng 1 và mang dấu trừ nếu nó có giá trị bằng 0.**

Ví dụ số 1111 khi không mang dấu đổi sang hệ 10 sẽ là:

$$1111_2 = 1*2^3+1*2^2+1*2^1+1*2^0 = 15_{10}$$

còn khi mang dấu sẽ là:

$$1111_2 = -1*2^3+1*2^2+1*2^1+1*2^0 = -8+4+2+1 = -1_{10}$$

Với mã nhị phân 4 bit chúng ta vẫn biểu diễn được 16 chữ số của hệ thập phân, khi đó số 4 bit lớn nhất sẽ là $0111_2=7_{10}$ và số nhỏ nhất sẽ là $1000_2 = -8_{10}$.

Với quy định dấu như trên có thể xuy ra số nhị phân 8 bit dương lớn nhất là:

$0111\ 1111_2 = 127_{10}$, và số nhỏ nhất là $1000\ 000_2 = -128_{10}$,

còn với mã 16 bit thì số dương lớn nhất là:

$$0111\ 1111\ 1111\ 1111_2 = 32767$$

và số âm nhỏ nhất là:

$$1000\ 0000\ 0000\ 0000_2 = -32768.$$

Với quy định này khi đổi số từ hệ 2 sang hệ 10 ta sẽ nhận được các giá trị khác đi so với bảng 1.2. (xem bảng 1.4)

Bảng 1.4 : Số nhị phân có dấu 4 bit

Hệ 2	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Hệ 10	0	1	2	3	4	5	6	7	-8	-7	-6	-5	-4	-3	-2	-1

Chú ý :

Với mã nhị phân 4 bit số nguyên lớn nhất có thể biểu diễn là 7 như vậy các số lớn hơn 7 cần phải biểu diễn bằng mã nhị phân lớn hơn 4 bit

Ví dụ : số 1001 không dấu đổi sang hệ 10 là bằng 9, khi biểu diễn có dấu thì đó là -7, còn số 9 lúc này sẽ phải biểu diễn bằng mã nhị phân 5 bit 01001, tương tự số 15 muốn biểu diễn có dấu thì phải là 0 1111.

VI. CÁC PHÉP TÍNH TRONG HỆ NHỊ PHÂN

1. Phép cộng

Khi thực hiện các phép tính trong hệ nhị phân cần lưu ý việc chúng ta sử dụng mã 4,8,16 hay 32 bit. Nguyên tắc cộng như sau:

$$0 + 0 = 0;$$
$$1 + 0 = 1;$$
$$0 + 1 = 1;$$

1 + 1 = 0 nhớ 1 lên bit trên của số trừ

Ví dụ: Cộng số với mã 8 bit:

```
 0000 1001
 0001 1001
 ----------
 0010 0010
```

2. Phép trừ

Phép trừ trong hệ nhị phân tuy không được định nghĩa song có thể thực hiện phép trừ thông qua phép cộng giữa số bị trừ với số đối của số trừ (số này còn được gọi là **số bù hai** của số trừ) ví dụ:

$$4 - 2 = 4 + (-2) = 2$$

Số bù hai của một số thực chất là số đối của số đó trên trục số. Như vậy nếu số ban đầu là số dương thì đối của nó sẽ là số âm và ngược lại.

Các số bù trong hệ đếm nhị phân được định nghĩa như sau :

**** Số bù 1 của một số nhị phân là số được hình thành từ số ban đầu bằng cách đảo các giá trị 0 thành 1 và 1 thành 0.***

Ví dụ số ban đầu là 1001 (khi có xét dấu bằng -7). Số bù 1 của 1001 là 0110.

**** Số bù 2 của một số nhị phân được tạo thành bằng cách lấy số bù 1 cộng thêm với 1.***

Ví dụ số bù 2 của 1001 sẽ là 0110 + 1 = 0111

Như vậy $1001_2 = -7_{10}$ và số bù 2 của 1001 là số 0111_2 là bằng $+7_{10}$.

Hoặc $0101_2 = 5_{10}$, bù 1 = 1010_2, bù 2 = 1010 +1 = 1011 = $-2^3+0+2^1+2^0 = -5_{10}$

Chú ý:

- Khi thực hiện phép trừ số bị trừ, số trừ và kết quả đều phải cùng một khuôn nghĩa là cùng số bit

- Theo quy định đã nêu về cách biểu diễn số âm thì số -9 trong hệ đếm nhị phân không phải là 0111 mà là 1 0111. Nhưng vì số ban đầu (số 9) được biểu diễn bởi mã nhị phân 4 bit nên số đối cũng chỉ giữ lại 4 bit và vì vậy ta không thấy được số thể hiện dấu (số đầu tiên bên trái).

Để thực hiện phép trừ ta lấy số bị trừ cộng với số bù 2 của số trừ, nếu số nhớ cuối cùng lên bit trên vượt quá khuôn khổ của mã (4,8,...bit) thì ta bỏ qua.

Ví dụ thực hiện phép trừ:

1011 - 1001 (trong hệ 10 đây là phép trừ 11-9 = 2)

Số bù 2 của 1001 là 0111, vậy ta có:

```
  1011
+ 0111
-----------
1 0010
↑——— Số nhớ lên bít trên bỏ qua do đó ta có kết quả là 0010
```

($0010_2 = 2_{10}$).

Ghi chú : Khi tính toán trên giấy chúng ta có thể áp dụng cách trừ nhanh như sau:

0 - 0 = 0; 1 - 0 = 1; 1 - 1 = 0; 0 -1 = 1 nhớ 1 lên bit trên của số trừ

Ví dụ :

```
   1000 0010
-       1111
----------------------
   0111 0011
```

3. *Phép nhân*

Phép nhân thực hiện theo nguyên tắc:

$$0 \times 0 = 0;\ 1 \times 0 = 0;\ 0 \times 1 = 0;\ 1 \times 1 = 1;$$

Khi nhân ta thực hiện nhân từng bit của số nhân (số thứ 2) với số bị nhân, kết quả của mỗi lần nhân viết dịch sang trái một vị trí sau đó cộng tất cả các giá trị trung gian.

Ví dụ: 1011 * 111

```
     1011
   * 111
 ------------
   1 0 1 1        dịch trái 0 bit
 1 0 1 1          dịch trái 1 bit
1 0 1 1           dịch trái 2 bit
-----------------
1 0 0 1 1 0 1
```

Có thể kiểm tra được rằng hai số đã cho trong hệ 10 là 11 và 7 và ta có 11×7 = 77, còn kết quả nhân trong hệ 2 cho ta:

$1011_2 * 111_2 = 100\ 1101$

$= 2^6+2^3+2^2+2^0$

$= 64+8+4+1 = 77$

4. *Phép chia*

Phép chia thực ra là phép trừ liên tiếp số bị chia cho số chia, mỗi một lần trừ ta cộng một đơn vị vào kết quả. Xét ví dụ trong hệ thập phân 36/9:

Phép trừ	Kết quả
36-9 = 27	1
27-9 = 18	1+1 = 2
18-9 = 9	2+1 = 3
9-9 = 0	3+1 = 4

Vậy 36 / 9 = 4

Do khuôn khổ bài giảng ở đây chỉ giới thiệu phép chia nguyên các số nhị phân. Áp dụng nguyên tắc đã nêu trên , ta thực hiện việc so sánh số bị chia với số chia. Nếu số bị chia lớn hơn hoặc bằng số chia thì thực hiện phép trừ và kết quả cộng thêm 1 đơn vị, tiếp tục quá trình cho đến khi kết quả phép trừ bằng 0. Xét ví dụ 1001/11, số bù 2 của 11 trong mã 4 bít là 1101

So sánh	Phép cộng số bù 2	Số dư	Kết quả
1001 > 11	1001+1101 =	0110	1
0110 > 11	0110+1101 =	0011	1+1 = 10
0011 = 11	0011+1101 =	0000	10+1 = 11

Có thể kiểm nghiệm phép chia trên trong hệ thập phân

$1001_2 = 9_{10}$; $11_2 = 3_{10}$ 1001/11 = 11 nghĩa là 9/3 = 3.

Trong trường hợp chia trên giấy chúng ta cũng có thể áp dụng nguyên tắc chia đã biết trong hệ đếm cơ số 10. Tuy nhiên đây không phải là cách mà máy tính thực hiện nên chúng tôi không khuyên bạn đọc sử dụng. Trở lại ví dụ trên theo cách chia đã biết trong hệ thập phân:

Đầu tiên lấy 100 chia cho 11 được 1, lấy 100 trừ 11 ta được 001, hạ tiếp số 1 xuống ta có 0011, chia tiếp 0011 cho 11 được 1 dư 0 vậy kết quả phép chia là 11

```
100 1   | 11
 11     | 11
--------
001 1
 - 1 1
--------
   00
```

VII. KHÁI NIỆM VỀ MÃ VÀ MÃ HOÁ

Con người lưu trữ dữ liệu thông qua việc sử dụng các chữ cái, chữ số và các ký tự toán học, đó là quá trình mã hoá thông tin. Máy tính phải thực hiện những yêu cầu của con người và phải trả lời những câu hỏi đặt ra bằng các mã thông dụng để con người có thể hiểu, vì vậy cần phải có một bộ ký tự được mã hoá dưới dạng nhị phân làm cơ sở để truyền các mệnh lệnh cho máy đồng thời lại phải biến đổi các kết quả trả lời của máy dưới dạng nhị phân thành các mã thông dụng. Hiện nay nhiều nước trong đó có Việt Nam đang sử dụng bộ mã truyền tin tiêu chuẩn của Hoa Kỳ với tên gọi là ASCII (American Standard Code for Information Interchange). Năm 1994 Bộ Khoa học Công nghệ và Môi trường đã công bố bộ mã

chuẩn quốc gia Việt Nam. Đây là cơ sở để các nhà tin học thiết kế các phần mềm ứng dụng sau này. Các ký tự trong bộ mã được đánh số từ 0 đến 255, nghĩa là có tất cả 256 ký tự. Thông thường người ta chỉ dùng 254 ký tự, vị trí 0 và 255 được để trống. Sở dĩ chỉ có 256 ký tự là vì các thế hệ máy vi tính đầu tiên dùng mã nhị phân 8 bit , số nhị phân lớn nhất 1111 1111_2 chính là 255. Trong bảng mã ASCII 128 ký tự đầu là cố định, các ký tự từ 128 đến 255 được gọi là các ký tự mở rộng, những ký tự này có thể được thay đổi tuỳ thuộc vào mục đích sử dụng hoặc ngôn ngữ của nước sử dụng. Tiếng Việt ngoài các phụ âm và nguyên âm còn có các bán nguyên âm và một số dấu đi kèm theo các nguyên âm, những ký tự này trong tiếng Anh không có vì vậy dùng mã ASCII ta không thể viết được tiếng Việt có dấu. Để khắc phục nhược điểm đó nhiều tác giả đã xây dựng các chương trình soạn thảo tiếng Việt như VNI, BKED, VIETRES, FREECODE, ABC, VIETKEY, UNICODE... trong đó phần lớn các ký tự tiếng Việt trùng với ký tự tiếng Anh được giữ nguyên, phần khác nhau được thiết kế bổ sung vào vùng mã ký tự mở rộng. Ngoài các chương trình soạn thảo văn bản chuyên dụng đã nêu trên còn một số chương trình thay đổi mã bàn phím và thường trú trong bộ nhớ của máy như VN.COM, TBK.COM... Các chương trình này có thể dùng kèm theo các phần mềm ứng dụng khác như FOXBASE, FOXPRO, QUATRO,... Hiện nay chúng đã trở nên lỗi thời nên không còn ý nghĩa sử dụng. Việc đưa tiếng Việt vào các phần mềm ứng dụng đã tạo thuận lợi rất nhiều cho việc quản trị dữ liệu, tuy nhiên việc chuyển đổi, in ấn vẫn còn gặp một số hạn chế như một số ký tự khi chuyển đổi không đúng dạng mong muốn...

1. Khái niệm mã hoá

Mã hoá là quá trình gán cho một nhóm đối tượng một nhóm chữ số tương ứng. Như vậy mã hoá thực chất là một quá trình chuyển giữa ngôn ngữ này sang ngôn ngữ khác. Ví dụ chúng ta gán cho 15 chữ cái đầu tiên các số tương ứng như sau:

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Vậy để viết chữ "khong" chúng ta có thể viết dưới dạng mã:

khong = 11 8 15 14 7

Tương ứng chữ " khong " trong hệ nhị phân sẽ là:

1011 1000 1111 1110 0111

2. Khái niệm về bảng mã

Bảng mã là bảng liệt kê tất cả các đối tượng của một ngôn ngữ với các giá trị mã hoá gán cho nó. Các giá trị này chính là số thứ tự của ký tự trong bảng mã. Dưới đây giới thiệu một số nhóm ký tự và mã của chúng trong bảng ASCII.

Từ số 1 đến 31 là các ký tự đặc biệt

Từ số 32 đến 47 là các ký tự ! " # $ %...

Từ số 48 đến 57 là các ký tự 0 1 2 3 4 5 6 7 8 9

Từ số 65 đến 90 là các ký tự A B C D .... Z

Từ số 97 đến 122 là các ký tự a b c d ..... z

Do hệ đếm cơ sở 16 dùng các ký tự A, B, C, D, E, F làm chữ số nên để phân biệt các ký tự với số chúng ta dùng cặp dấu nháy ' ', ví dụ 1A là số 1A trong hệ 16, còn '1A' là hai ký tự giống như hai chữ cái.

Chú ý:

Trong bảng mã một ký tự được quy định là lớn hơn ký tự khác khi số thứ tự của nó là lớn hơn, ví dụ b > a, z > t,... Với cách quy ước trên các chữ cái thường là lớn hơn các chữ cái in a > A.

VIII. TỔNG QUAN VỀ ĐẠI SỐ LÔGIC (ĐẠI SỐ BOOLE)

Đại số lôgic còn được gọi là đại số Boole hay đại số mệnh đề mang tên nhà toán học D.Boole (1815-1864) người đã đặt nền móng cho môn đại số này.

Nghiên cứu chương trình đại số ở phổ thông và đại học chúng ta đã biết một phương trình hay một biểu thức được hình thành từ các toán tử và toán hạng, trong đại số lôgic toán hạng sẽ là các mệnh đề lôgic còn toán tử sẽ là các phép toán đặc trưng của đại số lôgic như NOT, AND, OR....

Trong phạm vi giáo trình này chúng ta sẽ chỉ nghiên cứu một số phép tính lôgic và các vấn đề có liên quan đến việc thiết kế các mạch lôgic nhằm vào việc tính toán các phép toán lôgic và số học.

1. Định nghĩa mệnh đề

Mệnh đề là một câu nói hay một biểu thức có tính khẳng định hoặc phủ định một sự kiện nào đó. Tính chân lý của mệnh đề là ở chỗ mệnh đề chỉ có thể hoặc **đúng** hoặc **sai**. Nói khác đi một mệnh đề chỉ có thể nhận một trong hai giá trị đối nghịch nhau. Các giá trị đối nghịch này có thể là Đúng-Sai, Có-không, Sự thật-Giả dối, True-False, trong ngôn ngữ máy tính các giá trị đó là 1- 0.

Ví dụ: Băng lạnh hơn nước sôi.

$X^2 < 0$

Là các mệnh đề

Này, đi chơi đi.

Ôi, hoan hô bạn .

Không phải là các mệnh đề

Xét câu nói: Tháng 2 có 29 ngày

Sẽ có sự nhầm lẫn rằng câu nói này không phải là mệnh đề. Thực ra trong năm 2008 câu nói này hoàn toàn đúng nên nó là mệnh đề lôgic. Ba năm tiếp theo câu nói này sai nên nó cũng là mệnh đề. Tóm lại theo dòng thời gian từ quá khứ đến hiện tại và tương lai câu nói này luôn là mệnh đề lôgic.

Các mệnh đề có thể liên kết với nhau để tạo nên các mệnh đề mới, các mệnh đề liên kết này được gọi là ***biểu thức lôgic***. Giữa các mệnh đề tồn tại các phép toán cơ bản là: NOT, AND, OR

2. Biến và hàm

Biến lôgic là các đại lượng chỉ nhận giá trị trong tập D={0,1}

Hàm đơn trị F của n biến lôgic $x_1, x_2,... x_n$ được gọi là **hàm lôgic** nếu nó chỉ nhận các giá trị trong tập D. **Hàm lôgic** là một biểu thức liên kết các mệnh đề lôgic (các toán hạng) thông qua các toán tử lôgic (NOT, AND, OR). Giải một bài toán lôgic là quá trình xây dựng biểu thức hàm lôgic và tìm giá trị chân lý của hàm đó.

Nhận xét:

a. Với mọi tổ hợp n biến $x_1, x_2,... x_n$ hàm lôgic F có thể bằng 0, bằng 1 hoặc không xác định

b. Với một biến x chúng ta có hai tổ hợp giá trị là 0 và 1, với hai biến x, y ta có 4 tổ hợp giá trị là (0,0), (0,1), (1,0), (1,1), còn với 3 biến x,y,z chúng ta sẽ có 8 tổ hợp giá trị (0,0,0), (0,0,1), ... (1,1,1) . Tổng quát với hàm lôgic n biến ta có thể hình thành 2^n tổ hợp giá trị của các biến.

Người ta đã chứng minh được rằng với n biến lôgic có thể tìm được 2^{2^n} hàm lôgic nhận các giá trị trong tập D.

Ví dụ: với n = 1 nghĩa là chỉ có 1 biến x, ta có hai tổ hợp giá trị của x là x=0 và x=1 (vì $2^n = 2^1 = 2$) và có 4 hàm lôgic $F_1(x)$, $F_2(x)$, $F_3(x)$, $F_4(x)$ mà giá trị của chúng nằm trong D (bảng 1.5).

Bảng 1.5

x \ F	F_1	F_2	F_3	F_4
0	0	0	1	1
1	0	1	0	1

Nhìn vào bảng 1.5 ta có nhận xét như sau:

Với mọi giá trị của biến x giá trị của hàm F có thể rơi vào 1 trong 4 trường hợp :

a. Luôn luôn bằng 0 (hàm F_1)

b. Luôn luôn bằng 1 (hàm F_4)

c. Luôn luôn bằng giá trị của x (hàm F_2)

d. Luôn luôn ngược giá trị của x (hàm F_3)

Phép toán một ngôi tác động đến biến x khiến cho giá trị của hàm một biến F luôn ngược với giá trị của x được gọi là phép "**đảo**" hay phép "**phủ định**" và được ký hiệu là NOT.

Như vậy F(x) = NOT(x) $\begin{cases} = 1 \text{ khi } x = 0 \\ = 0 \text{ khi } x = 1 \end{cases}$

Toán tử NOT còn được ký hiệu bởi nét (−) , NOT(x) = $\bar{x}$

Với 2 biến x và y, như đã nêu trên chúng ta có 4 tổ hợp giá trị :

(x=0, y=0); (x=0, y=1); (x=1, y=0); (x=1, y=1)

Từ đó có thể tìm được 16 hàm lôgic nhận các giá trị trong tập D (bảng 1.6)

Bảng 1.6

x y\F	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16
0 0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0 1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1 0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1 1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Nhận xét :

a. Với hàm lôgic đơn trị 2 biến chúng ta có thể hình thành 4 tổ hợp giá trị và tìm được 16 hàm lôgic ứng với từng bộ giá trị này .

b. Hàm F_2 nhận giá trị bằng 1 khi và chỉ khi cả x và y đều bằng 1. Phép toán lôgic hình thành nên hàm F_2 được gọi là **phép nhân lôgic** và ký hiệu là AND

c. Hàm F_8 nhận giá trị 0 khi và chỉ khi cả x và y đều bằng 0. Phép toán lôgic hình thành nên hàm F_8 được gọi là **phép cộng lôgic** và ký hiệu là OR

d. Hàm F_{10} nhận giá trị bằng 0 khi giá trị của x và y bằng nhau, bằng 1 khi giá trị của x và y khác nhau. Phép toán lôgic hình thành nên hàm F_{10} được gọi là **phép tương đương** (hay còn được gọi là phép "hoặc triệt tiêu") và được ký hiệu là XOR

e. Với hàm lôgic đơn trị 3 biến x,y,z số tổ hợp giá trị của biến sẽ là 8 và số hàm lôgic là 256 . Như vậy chỉ với 3 biến bài toán đã trở nên rất phức tạp do đó chúng ta phải tìm cách đơn giản hoá bài toán bằng cách biểu diễn hàm F thông qua một số hàm trung gian $f_1, f_2, ...f_m$.

3. Xác định giá trị chân lý của hàm lôgic

Trong một biểu thức lôgic các toán hạng nằm trong dấu ngoặc đơn sẽ có mức ưu tiên cao nhất, tiếp đó là đến toán tử NOT sau đó là toán tử AND cuối cùng là toán tử OR.

Để xác định giá trị chân lý của hàm lôgic chúng ta sử dụng một số kết quả sau đây đã được chứng minh trong đại số Boole :

a. Mọi hàm lôgic hai trị có n biến $F(x_1,x_2,...x_n)$ có thể nhận giá trị 1 tại một số tổ hợp biến và nhận giá trị 0 hoặc không xác định tại một số tổ hợp biến khác.

b. Mọi hàm lôgic đều có thể biểu diễn thông qua 3 toán tử NOT, AND, OR.

c. Mọi hàm lôgic n biến $F(x_1,x_2,...x_n)$ đều có thể biểu diễn được dưới một trong hai dạng chuẩn tắc sau đây :

* ***Dạng Mintec hay dạng chuẩn tuyển***

Với hàm lôgic n biến, giả sử có m vị trí ứng với các giá trị k_1, k_2, ... k_n tại đó hàm nhận giá trị bằng 1 khi đó có thể biểu diễn hàm dưới dạng chuẩn tuyển tức là dạng tổng của các toán hạng , mỗi toán hạng là một tích của các biến lôgic quy đổi.

$$F(x_1,x_2,...x_n) = \sum_{1}^{m} x_1^{k1} . x_2^{k2} ... x_n^{kn} \qquad (1.2)$$

$$F(k_1, k_2, ... k_n) = 1$$

Giá trị của các biến quy đổi x_i^{ki} ở vế phải của phương trình (1.2) được xác định tuỳ thuộc vào giá trị của các k_i, nếu :

$$ki = 0 \quad thì \quad x_i^{ki} = \overline{x_i} = NOT(x_i)$$

$$ki = 1 \quad thì \quad x_i^{ki} = x_i$$

Dạng Maxtec hay dạng chuẩn hội **(Dạng tích của các tổng)**

$$F(x_1,x_2,...x_n) = AND\ (OR\ x_i^{ki}) \qquad (1.3)$$

với $F(k1, k2... ki) = 0$

Giá trị của các toán hạng x_i^{ki} ở vế phải của phương trình (1.3) được xác định như sau:

Nếu : $ki = 0$ thì $x_i^{ki} = x_i$

còn nếu $ki = 1$ thì $x_i^{ki} = \overline{x_i} = NOT(x_i)$

Chú ý rằng trong cả hai trường hợp giá trị của ki lấy trong tập D [0,1].

Phương trình 1.2 được hiểu như sau: Một hàm lôgic n biến có thể được biểu diễn dưới dạng tổng (OR) của n toán hạng, mỗi toán hạng là một tích (AND) của các biến quy đổi (x_i^{ki}), trong đó giá trị của các ki được xác định sao cho khi thay vào phương trình ban đầu thì giá trị chân lý của hàm luôn luôn bằng 1.

Phương trình 1.3 khác phương trình 1.2 ở chỗ hàm F được xác định thông qua một tích (AND) của n toán hạng, mỗi toán hạng là một tổng (OR) của các biến.

Để thấy rõ hơn vấn đề ta xét một ví dụ:

Cho hàm F(x,y) = x XOR y, cần tìm các dạng Mintec và Maxtec của F.

Theo bảng 1.2 ta có F(0,0) = 1 và F(1,1) = 1 nghĩa là có hai cặp giá trị của k1 và k2 là (0,0) và (1,1) khiến cho hàm F luôn luôn bằng 1.

Khi $k1 = 0 \rightarrow x = \bar{x} = NOT(x)$,

$k2 = 0 \rightarrow y = \bar{y} = NOT(y)$

$k1 = 1 \rightarrow x = x$,

$k2 = 1 \rightarrow y = y$

Do đó ta có hàm Mintec:

$$F(x,y) = \underset{1}{\overset{n}{OR}}(AND\ x_i^{ki}) = (\bar{x}\ AND\ y) OR (\bar{x}\ AND\ y) \quad (1.4)$$

Theo bảng (1.2) có hai cặp giá trị (1,0) và (0,1) làm cho hàm F có giá trị 0 .

Khi $k1 = 1 \rightarrow x = \bar{\bar{x}}$

$k2 = 0 \rightarrow y = y$

$k1 = 0 \rightarrow x = \bar{\bar{x}}$

$k2 = 1 \rightarrow y = y$

Do đó ta có hàm Maxtec:

$$F(x,y) = \underset{1}{\overset{n}{AND}}(OR\ x_i^{ki}) = (\bar{\bar{x}}\ OR\ y)\ AND\ (\bar{x}\ OR\ y) \quad (1.5)$$

Với cách xây dựng các hàm Mintec và Maxtec, việc xác định giá trị chân lý của hàm F có thể được thay bằng việc xác định giá trị chân lý của biểu thức vế phải của các phương trình (1.4) hoặc (1.5). Nói khác đi giá trị chân lý của một hàm n biến có thể được xác định thông qua n hàm đơn giản, giá trị chân lý của các hàm đơn giản này được xác định thông qua 3 toán tử NOT, AND, OR.

4. Rút gọn hàm lôgic

4.1 Rút gọn bằng phương pháp đại số

Để đơn giản cách viết hàm logic chúng ta sử dụng dấu chấm (.) thay cho phép AND, dấu + thay – cho phép OR và dấu - thay cho phép NOT. Khi đó chúng ta có một số hệ thức sau đây (bảng 1.7).

Bảng 1.7

1	$\bar{x} + x = 1$	7	$\bar{x} . x = x$
2	$x + 1 = 1$	8	$x . x = 0$
3	$\bar{x} + 1 = 1$	9	$x . 0 = 0$
4	$x + x = x$	10	$x+x.y = x$
5	$x + 0 = x$	11	$x+\bar{x} .y = x+y$
6	$x . 1 = x$	12	$(x+y).(x+z) = x+y.z$

Các tính chất của biểu thức lôgic

a. Tính giáo hoán

$x + y = y + x$; $x . y = y . x$

b. Tính kết hợp

$x + y + z = (x + y) + z = x + (y + z)$

$x . y . z = x . (y . z) = (x . y) . z$

c. Tính phân phối

$x . (y + z) = x . y + x . z$

$x + (y . z) = (x + y).(x + z)$

d. Định lý D. Morgan

$\overline{x + y} = \bar{x} . \bar{y}$; $x . y = \bar{x} + \bar{y}$

Ví dụ : Rút gọn hàm lôgic sau đây : $F(x,y,z) = x.y.\bar{z} + x.y.z + x.\bar{y} . \bar{z}$

Áp dụng hệ thức số 4 ta có : $F(x,y,z) = x.y.\bar{z} + x.y.z + x.\bar{y} . \bar{z} + x.y. \bar{z}$

Áp dụng tính phân phối ta có : $F(x,y,z) = x.y.(\bar{z} + z) + x.\bar{z} .(y + \bar{y})$

Áp dụng hệ thức số 1 ta có : $F(x,y,z) = x.y + x.\bar{z}$

4.2 Rút gọn bằng bìa Cacnô (Karnaugh)

Bìa Cacnô là một công cụ trực quan dùng để rút gọn hàm lôgic. Bìa Cacnô chỉ thuận tiện với những hàm mà số biến không nhiều (thường số biến nhỏ hơn 6).

Bìa Cacnô thực chất là một bảng biểu bao gồm các dòng và các cột, các cột biểu diễn một số tổ hợp của một số biến trong hàm, các dòng biểu diễn các tổ hợp của các biến còn lại. Sử dụng các toán tử NOT, AND, OR kết hợp các nhóm biến trên hàng và cột ta được các tổ hợp biến mới, chúng có thể là toán hạng của hàm

đã cho hoặc cũng có thể không. Các tổ hợp biến bố trí sao cho các biến đứng gần nhau có giá trị khác nhau. Cần chú ý rằng cột cuối cùng được coi là đứng cạnh cột đầu tiên, dòng cuối cùng được coi là xếp cạnh dòng đầu tiên.

Nếu tổ hợp các biến trong hàng và cột cho ta một toán hạng của hàm ban đầu thì tại vị trí giao nhau của hàng và cột ta viết số 1, các vị trí khác để trống. Tổ hợp các toán hạng trong bìa Cacnô theo hàng hoặc cột ta có thể dễ dàng tìm được dạng rút gọn tối thiểu của hàm.

Xét ví dụ :

$$F(x,y,z,t) = \bar{x}\,.\bar{y}\,.\bar{z}\,.\bar{t} + \bar{x}\,.\bar{y}\,.z.\bar{t} + \bar{x}\,.y.\bar{z}\,.t + x.\bar{y}\,.\bar{z}\,.\bar{t} + x.\bar{y}\,.\bar{z}\,.t + x.\bar{y}\,.z.\bar{t} + x.\bar{y}\,.z.t + x.y.\bar{z}\,.t$$

Hàm F bao gồm 4 biến x,y,z,t , các tổ hợp của 2 biến x, y được lấy làm tiêu đề các cột còn tổ hợp của 2 biến z,t được lấy làm tiêu đề cho dòng.

zt \ xy	$\bar{x}\,.\bar{y}$ ①	$\bar{x}\,.y$ ②	$x\,.y$ ③	$x\,.\bar{y}$ ④
$\bar{z}\,.\bar{t}$	1			1
$\bar{z}\,.t$		1	1	1
$z.t$				1
$z.\bar{t}$	1			1

Tổ hợp các toán hạng ở cột 1 ta có $\bar{x}\,.\bar{y}\,.\bar{z}\,.\bar{t} + \bar{x}\,.\bar{y}\,.z.\bar{t} = \bar{x}\,.\bar{y}\,.\bar{t}$

Tương tự như vậy tổ hợp các toán hạng ở cột 4 ta được $x.\bar{y}$

Tổ hợp các toán hạng ở hai ô giữa bảng ta có $y.\bar{z}\,.t$

Vậy hàm F sẽ có dạng rút gọn tối thiểu là:

$$F(x,y,z,t) = \bar{x}\,.\bar{y}\,.\bar{t} + x.\bar{y} + y.\bar{z}\,.t$$

5. Các toán tử lôgic - ký hiệu mạch điện và sơ đồ công tắc

Các toán tử sử dụng trong đại số lôgic bao gồm: Not, And, Or, Xor...

a. Toán tử OR (Hoặc- còn gọi là cộng lôgic) :

***Xét* câu nói sau:**

Nếu trời đẹp hoặc có xe đón thì lớp đi tham quan.

Câu nói này thể hiện một nguyên tắc là việc đi tham quan sẽ được thực hiện (true) khi một trong hai yếu tố "trời đẹp" , "xe đón" xảy ra (true).

Ký hiệu mệnh đề "trời đẹp" là A, "xe đón" là B và "đi tham quan" là C thì chúng ta có thể chuyển hóa câu nói thành một biểu thức lôgic.

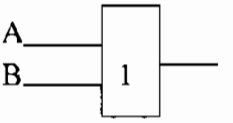
A OR B = C

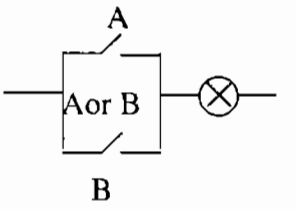
Nếu đúng được ký hiệu là 1 thì sai ký hiệu là 0, vì A và B có thể nhận một trong hai giá trị 0 và 1 nên có 4 khả năng xảy ra đối với C (Hình 1.1). Các giá trị của A, B và C được viết trong một bảng gọi là Bảng chân lý của các toán tử lôgic (Truth Table). Từ bảng chân lý có thể kết luận là:

Hai mệnh đề lôgic A và b kết hợp bởi toán tử OR cho kết quả sai (0) khi và chỉ khi cả A và B đều sai (0)

Bảng chân lí Kí hiệu mạch Sơ đồ công tắc

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1





Hình 1.1

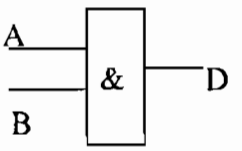
b. Toán tử AND (Và - còn gọi là nhân lôgic) (hình 1.2)

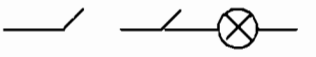
Trở lại ví dụ trong mục a nhưng thay vì dùng từ "Hoặc" chúng ta dùng từ "Và". Điều này có nghĩa là việc đi tham quan chỉ xảy ra khi đồng thời hai điều kiện "trời đẹp' và "xe đón" là đúng. Từ bảng chân lý có thể kết luận là:

Hai mệnh đề lôgic A và b kết hợp bởi toán tử OR cho kết quả sai (0) khi và chỉ khi cả A và B đều sai (0)

Bảng chân lí Kí hiệu mạch Sơ đồ công tắc

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1





A and B

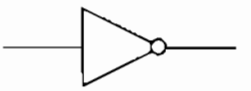
đèn chỉ sáng khi cả A và B đóng

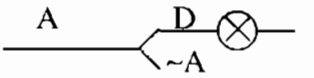
Hình 1.2

c- Mạch NOT (đảo hay còn gọi là phủ định):

Bảng chân lí Kí hiệu mạch Sơ đồ công tắc

A	NOT A
0	1
1	0





Hình 1.3

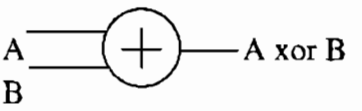
d- Mạch XOR (Hoặc triệt tiêu) :

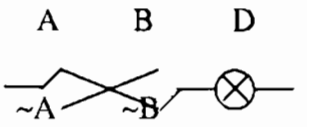
Từ bảng chân lý có thể kết luận là:

Hai mệnh đề lôgic A và B kết hợp bởi toán tử XOR cho kết quả sai (0) khi A và B giống nhau, còn lại cho kết quả đúng (1).

Bảng chân lí Kí hiệu mạch Sơ đồ công tắc

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0





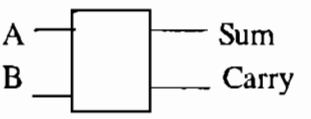
đèn sáng khi A và ~B hoặc khi ~A và B

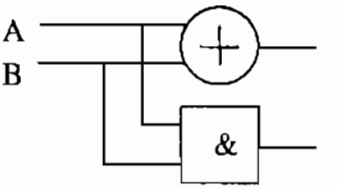
Hình 1.4

e- Mạch ADD (Mạch cộng hai số nhị phân)

Bảng chân lí Kí hiệu mạch Sơ đồ công tắc

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1





Khi cửa A=1 và cửa B=1 thì cửa Sum=0 cửa Carry=1. Vậy 1+1 = 0 nhớ 1

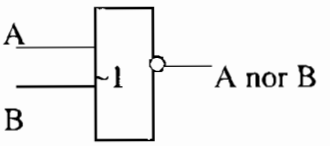
Hình 1.5

f- Mạch NOR (OR-NOT) : Đây là mạch đảo của mạch OR

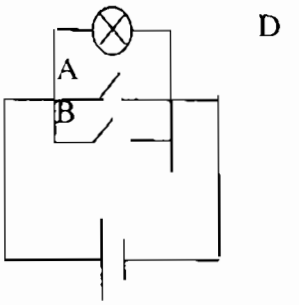
Bảng chân lí

A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

Kí hiệu mạch



Sơ đồ công tắc



Hình 1.6

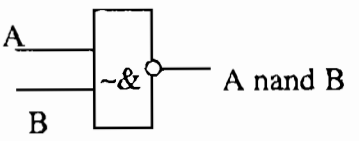
Khi A hoặc B ở vị trí đóng (A=1 hoặc B=1) hoặc cả hai công tắc ở vị trí đóng (A=1 và B=1) thì đèn bị tắt (Đ=0) vì các công tắc làm đèn bị đoản mạch. Đèn sáng khi A và B đều ngắt.

g - Mạch NAND *(AND-NOT)* : Đây là mạch đảo của mạch AND

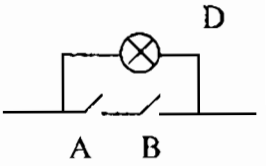
Bảng chân lí

A	B	A nand B
0	0	1
0	1	1
1	0	1
1	1	0

Kí hiệu mạch



Sơ đồ công tắc



Hình 1.7

Khi một trong hai công tắc ở vị trí ngắt (A=0 hoặc B=0) hoặc cả hai công tắc đều ngắt thì đèn sáng (Đ=1), còn khi cả hai công tắc đều đóng thì đèn tắt vì công tắc làm đoản mạch.

Hai mạch NAND, NOR có đặc điểm là:

Mạch NAND là mạch tổ hợp từ mạch AND và mạch NOT trong đó toán tử AND thực hiện trước, toán tử NOT thực hiện sau. Mạch NOR là tổ hợp của OR và NOT với OR thực hiện trước và NOT thực hiện sau.

6. Ứng dụng của các mạch lôgic trong máy tính

Các mạch lôgic có rất nhiều ứng dụng trong tin học, chẳng hạn ứng dụng trong việc thực hiện các phép tính hệ 2. Bốn phép tính số học trong hệ 2 đều có thể qui về một phép toán số học duy nhất là phép cộng như sau :

- phép trừ chuyển thành phép cộng với số bù 2 ,
- phép nhân chuyển thành phép cộng với các số dịch trái (Shift left - SHL)
- phép chia chuyển thành phép cộng với mã bù 2 của số dịch phải (Shift right - SHR)z.

Vì vậy trong phần này ta xét hai mạch cơ bản nhất đó là mạch cộng 2 số nhị phân và mạch chuyển đổi sang mã số bù 2.

6.1 Mạch cộng 2 số nhị phân

* Nguyên tắc: Thực hiện cộng từng cặp bit của 2 số theo trình tự từ phải qua trái, các cặp bit này phải có cùng vị trí (vị trí được đánh số bắt đầu từ số 0 từ phải qua trái). Với mỗi phép cộng ta có 2 số: Sum (tổng) và Carry (số nhớ).

Cần chú ý rằng:

1+1 = 0 nhớ 1

0+1 = 1 nhớ 0

1+0 = 1 nhớ 0

Các số Sum và Carry này được xem là đầu vào cho lần cộng thứ hai theo nguyên tắc lấy số Carry của bit thấp cộng với số Sum của bit cao (hình 1.8) ví dụ: cho A = 0101, B = 0110

Thực hiện phép cộng A + B

* Tầng thứ nhất:

Đặt bit A_0,B_0 vào vào cửa a_0,b_0 của mạch cộng (A_0=1, B_0=0) → 1+0 = 01

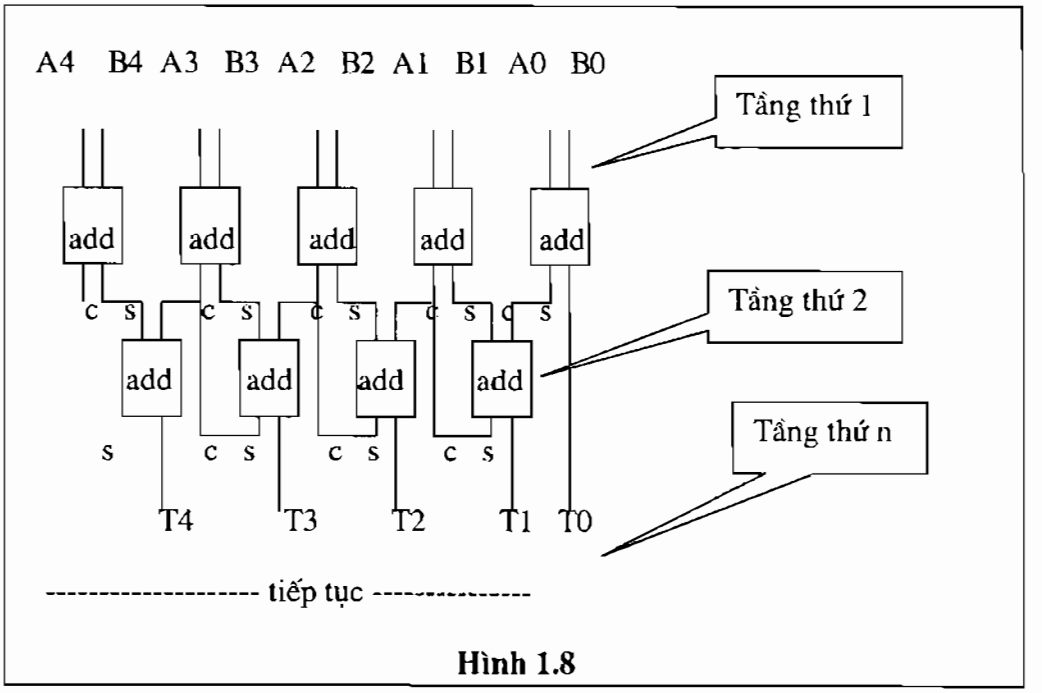
Đặt bit A_1,B_1 (0,1) vào vào cửa a_1,b_1 → 0+1 = 01

Đặt bit A_2,B_2 (1,1) vào vào cửa a_2,b_2 → 1+1 = 10

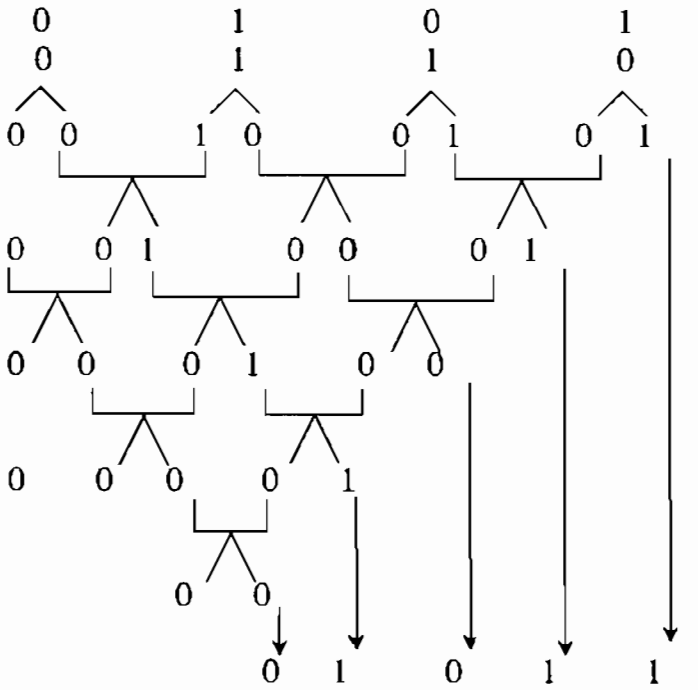
Đặt bit A_3,B_3 (0,0) vào vào cửa a_3,b_3 → 0+0 = 00

* Tầng thứ hai: lấy số Carry ở cửa 0 cộng với số sum ở cửa 1, lấy số Carry ở cửa 1 cộng với số sum ở cửa 2... (Hình 1.9)

Như vậy nếu cần cộng số nhị phân 8 bit ta cần 8 cửa vào đánh số từ 0 đến 7 và sẽ có 8 tầng chuyển tiếp. Trong ví dụ đã nêu ta chỉ cần 4 cửa vào và do đó có 4 tầng chuyển tiếp. Ở cưả ra của mạch cộng cho kết quả 1011.



Hình 1.8



Hình 1.9

6.2 Mạch tạo mã bù 2

Như đã trình bày mã bù 2 là số đối của số nhị phân đã cho (xem phần biểu diễn số âm), để có mã bù 2 ta tạo mã bù 1 bằng cách đổi số 0 thành 1 và số 1 thành 0. Lấy mã bù 1 cộng với 1 ta có mã bù 2.

Ví dụ: xét các số trong bảng sau:

Số ban đầu	1101	1010*10*	10*1000*
Mã bù 1	0010	0101101	010111
Mã bù 2	0011	0101*10*	01*1000*

Qua bảng có thể nêu một số nhận xét như sau:

* Bit thấp nhất (bit đầu tiên bên phải) của số đã cho khi chuyển sang bù 2 vẫn giữ nguyên.

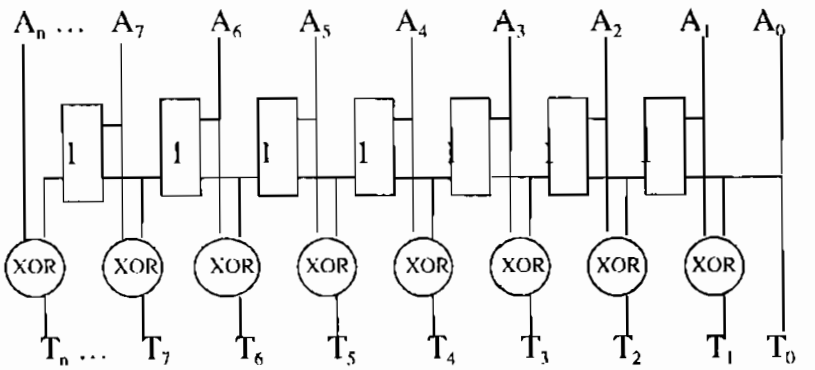
* Nếu bit thấp nhất là số 1 thì bit đầu tiên của mã bù 2 cũng vẫn là 1, khi đó để có mã bù 2 ta chỉ cần chuyển đổi các bit còn lại sang bù 1.

* Nếu bit thấp nhất là số 0 thì xét các bit tiếp theo (theo chiều từ phải qua trái) cho đến khi gặp bit đầu tiên có giá trị là 1. Giữ nguyên các bit từ bit đầu tiên đến bit bằng 1 đó và đổi các bít còn lại ở bên trái sang bù 1 ta có mã bù 2 (các số viết nghiêng trong bảng).

Trong bảng ta có số 101000

Giữ nguyên bốn bit 1000 tính từ trái qua phải, còn lại số 10 ta chuyển sang bù 1 thành 01. Vậy mã bù 2 sẽ là 011000.

Trên hình vẽ mô tả một cách hình thức việc đổi số A = (A_0, A_1,....,A_n) thành số mã bù 2, T = (T_0,T_1,...,T_n). Cần chú ý rằng giá trị n ở đây không phải là tùy ý mà phụ thuộc vào bộ vi xử lý của máy tính, thường là 8, 16, 32, 64 bit.



Bit đầu tiên A_0 được chuyển thành bit đầu tiên T_0 = của mã bù 2, các bit còn lại tùy thuộc vào giá trị của bit đầu mà được chuyển thẳng hoặc chuyển thành mã bù 1. Để chuyển thành mã bù 1 ta chỉ việc đưa bít đó vào mạch Xor.

Ví dụ số 1101 trong bảng trên:

bit A_0 = 1 chuyển thẳng, các bit còn lại

A_1 xor 1 = 1;

A_2 xor 1 = 0,

A_3 xor 1 = 0 vậy mã bù 2 của 1101 bằng 0011

7. Biểu thức lôgic

Đại số lôgic không nghiên cứu văn phạm của ngôn ngữ mà chỉ quan tâm đến giá trị chân lý của mệnh đề. Mỗi mệnh đề được xem như một toán hạng, các toán hạng liên kết bởi các toán tử để hình thành nên các biểu thức logic hay còn gọi là hàm lôgic.

7.1. Định nghĩa

Biểu thức lôgic là một biểu thức toán học mà toán hạng là các mệnh đề lôgic còn toán tử là các toán tử lôgic kết hợp với các toán tử số và toán tử so sánh.

7.2. Các ví dụ

Ví dụ 1: Các biểu thức so sánh sau:

$Sin^2a + cos^2a > 2$ cho kết quả 0 (sai - False)

$Sin90^o = 1$ cho kết quả 1 (đúng -True)

7.3. Thứ tự thực hiện

Trong một biểu thức lôgic thứ tự ưu tiên các toán tử là:

Toán hạng	(.....)	Not	And	Or
Mức ưu tiên	1	2	3	4

Ví dụ 2: Hãy chuyển hóa câu ca dao sau thành một biểu thức lôgic và xác định giá trị chân lý của biểu thức lôgic đó

"Bao giờ Chạch đẻ ngọn đa, Sáo đẻ dưới nước, thì ta lấy mình".

Có thể xem như đây là câu trả lời của một cô gái đối với lời cầu hôn của một chàng trai.

Ký hiệu các mệnh đề :

"*Chạch để ngọn đa*" là A. "*Sáo để dưới nước*" là B. "*Thì ta lấy mình*" là C.

Chúng ta chưa có cơ sở để kết luận rằng hai mệnh đề A và B phải kết hợp bởi toán tử nào do đó phải xét cả hai toán tử And và Or. Vì cả hai mệnh đề A và B đều nhận giá trị 0 (sai) do đó căn cứ vào bảng chân lý ta có :

A OR B = C = 0

A AND B = C = 0

Trong cả hai trường hợp mệnh đề C đều nhận giá trị 0. Rõ ràng là với một câu trả lời rất tế nhị song cô gái đã gửi một thông điệp hết sức rõ ràng và kiên quyết, trong bất kỳ trường hợp nào cũng không tồn tại việc "Ta lấy mình".

4. Các toán tử AND, OR, XOR trong phép tính với số nhị phân

Khi thực hiện các toán tử AND, OR, XOR trên các số nhị phân chúng ta thực hiện theo từng bit, các bit này phải có cùng vị trí tính từ phải qua trái. Kết quả của phép AND, OR, XOR trên số nhị phân vẫn căn cứ vào bảng chân lý của các toán tử đã biết. Đây là nguyên tắc mà Pascal dùng để pha màu khi làm việc trong chế độ đồ họa thông qua các thủ tục ANDPUT, ORPUT, XORPUT.

Ví dụ nếu dùng ba thủ tục trên pha hai màu trắng mã màu 15 (1111) và nâu mã màu 6 (110) ta được các màu lần lượt là nâu (110), trắng (1111) và xanh lam sáng (1001) (xem bảng 1.12)

Bảng 1.12

```
      1111          1111          1111
AND    110    OR     110    XOR    110
    --------      --------      --------
      0110          1111          1001
```

Chú ý:

1. Trong đại số Boole không được phép đơn giản các toán hạng ở hai vế phương trình như trong đại số cao cấp. Ví dụ:

A and B = A and C; không thể đơn giản A ở cả hai vế để còn lại B = C, điều này có thể kiểm tra qua các giá trị cụ thể A = 0, B = 0, C = 1.

A or B = A or C, nếu lấy A = 0, B = 0, C = 1 thì rõ ràng là không thể đơn giản hai vế cho A vì nếu thế ta thấy ngay B # C;

2. Để tiện cho việc viết các hàm Boole chúng ta thay thế việc viết toán tử AND bằng dấu nhân (.) và toán tử OR bằng dấu cộng (+) nhưng không nên hiểu đó là các toán tử (+) và (.) như trong đại số thông thường.

Chương 2

NGUYÊN LÝ CẤU TẠO MÁY VI TÍNH

Tất cả máy tính PC hiện nay đều được cấu tạo bởi hai phần: Phần cứng và phần mềm. Phần cứng bao gồm các linh kiện điện tử, vi mạch lắp ráp nên các bộ phận của máy tính như bàn phím, màn hình, thân máy... Phần mềm là các chương trình được cài đặt vào máy phục vụ các yêu cầu riêng của từng người sử dụng (hình 2.1).

I. PHẦN CỨNG

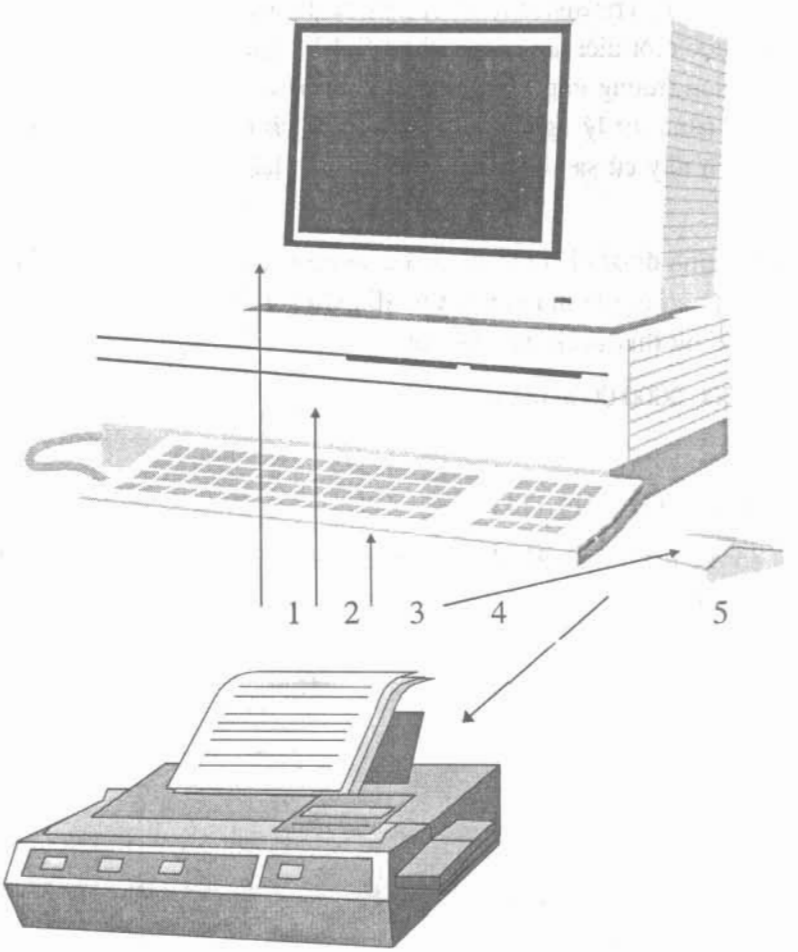
Máy vi tính (Micro Computer) là hệ thống thiết bị điện tử được lắp ráp bởi các linh kiện điện tử và mạch vi xử lý. Máy vi tính hoạt động theo chương trình nhằm xử lý, lưu trữ và trao đổi thông tin.

Thuật ngữ "Máy vi tính" được dùng với mục đích để chỉ các máy tính có kích thước nhỏ vì so với thế hệ các máy tính điện tử thì quả thật kích thước của chúng đã được thu bé một cách đáng kể. Tuy nhiên thuật ngữ này ngày càng trở nên không phù hợp khi các máy tính xách tay xuất hiện. Ngày nay người ta sử dụng thuật ngữ Personal Computer viết tắt là PC để chỉ các máy tính dùng cho cá nhân. Do tính lịch sử trong sách này chúng ta vẫn dùng thuật ngữ "Máy vi tính".

Nhìn bề ngoài, máy vi tính gồm các bộ phận sau đây: (hình 2.1)

1. Màn hình (Monitor hay Screen);
2. Case (phần thân máy bao gồm bộ vi xử lý- Central Processing Unit - CPU, đĩa cứng,...);
3. Bàn phím (Keyboard - Console);
4. Thiết bị chuột (Mouse);
5. Máy in (Printer).

Máy tính mà chúng ta đang dùng hiện nay thuộc thế hệ thứ tư, tuỳ theo cấu trúc linh kiện vi xử lý mà người ta phân thành loại 286, 386, 486, 586. Các máy thế hệ 286 bộ vi xử lý dùng mã 16 bit nhưng Bus dữ liệu (cáp truyền dữ liệu) chỉ dùng 8 bit. Sang thế hệ 386 với vi xử lý 32 bit, hãng IBM cũng như một số hãng nổi tiếng khác trên thế giới đã sử dụng hai loại Bus dữ liệu , loại 386 SX dùng Bus dữ liêu 16 bit, loại 386 DX dùng Bus dữ liệu 32 bit. Những máy tính sản xuất gần đây là loại 486 thì vi xử lý dùng mã 64 bit và cũng gồm hai loại SX và DX.



Hình 2.1

Những máy tính dành cho các cá nhân sử dụng được gọi chung là máy tính cá nhân (Personal Computer) hay còn gọi là máy vi tính (Micro Computer). Những máy tính dùng làm máy cái trong mạng, hay tại các nút mạng được gọi là máy mẹ và thường là các máy mini. Cấu trúc bên trong của máy vi tính gồm các khối cơ bản sau đây: Khối xử lý, các thiết bị vào ra, bộ nhớ (hình 2.2).

II. BỘ NHỚ CỦA MÁY VI TÍNH

Bộ nhớ của máy vi tính được chia thành hai loại: Bộ nhớ trong và bộ nhớ ngoài. Bộ

nhớ trong ngày nay được chế tạo dưới dạng các Chip vi mạch nhớ với dung lượng mỗi Chip có thể lên đến 128 MB. Bộ nhớ ngoài được chế tạo dưới dạng các đĩa từ hoặc đĩa laser và dung lượng của chúng có thể lên đến hàng trăm GB. Các linh kiện để ghi nhớ của bộ nhớ trong chính là các Transistor nối với một tụ điện có điện dung cỡ vài chục pF. Các mạch này ghi được một điện áp tương ứng với 1 bit (giá trị 1 nhị phân), sau vài ms mạch sẽ phóng hết điện (tương ứng với giá trị 0 nhị phân). Như vậy để lưu giá trị 1 nhị phân trong suốt quá trình xử lý người ta dùng một mạch tự động gọi là mạch làm tươi lại RAM động, mạch này cứ sau khoảng 2 ms lại nạp lại điện cho mạch nhớ một lần.

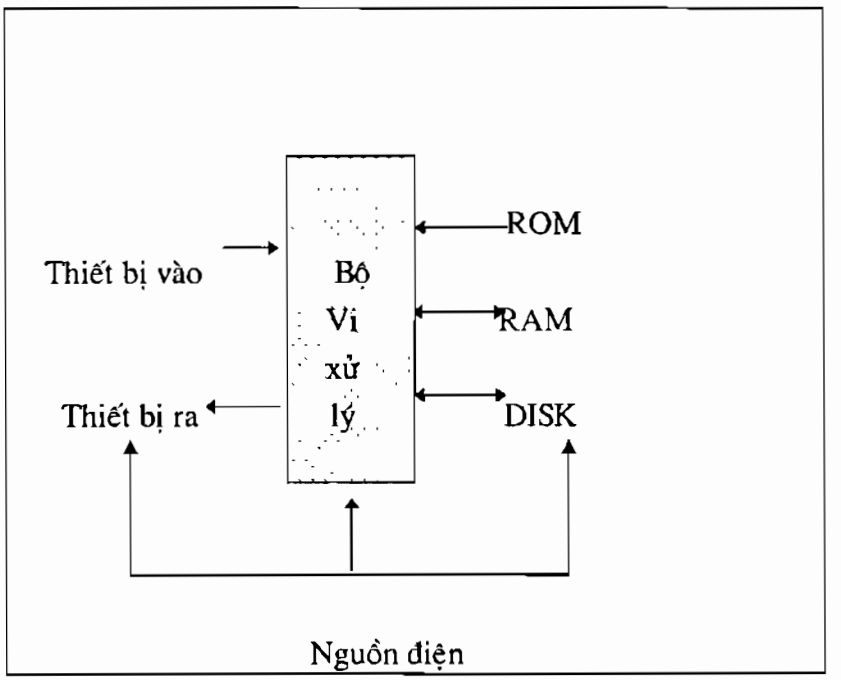
Bộ nhớ của máy vi tính được chia ra thành các ô nhớ, mỗi ô nhớ có dung lượng 1 byte và được gán cho một số gọi là địa chỉ ô. Các địa chỉ ô bắt đầu từ 0. Trong những máy dùng mã nhị phân 16 bit thì địa chỉ bắt đầu là:

0000 0000 0000 0000

và kết thúc là:

1111 1111 1111 1111.

Như vậy ta chỉ có thể đánh địa chỉ các ô nhớ từ 0 đến 65535 hay nói khác đi bộ nhớ chỉ có thể tối đa là 64 Kbyte.



Hình 2.2

Để thiết kế những bộ nhớ lớn hơn 64 KB, người ta phải dùng hai đại lượng cho mỗi địa chỉ ô nhớ đó là địa chỉ đoạn (Segment) và vị trí tương đối của ô nhớ trong đoạn (Offset), mỗi đại lượng này được ghi trong một ô nhớ đặc biệt gọi là thanh ghi. Giá trị Segment được ghi trong thanh ghi CS (Code Segment), còn giá trị Offset được ghi trong thanh ghi IP (Instruction Point).

Xét một ví dụ tượng trưng:

Để tạo ra một bộ nhớ có dung lượng 1024 KB (1 MB) ta cần có 16 đoạn mỗi đoạn 64 K. Giả sử ta đánh số các đoạn đó từ 0 -15 và ghi địa chỉ ô nhớ theo hai tham số:

11:1111 1111 1111 1111

Địa chỉ trên cho ta biết ô nhớ nằm ở vị trí cuối cùng (địa chỉ 65535) trong đoạn thứ 3 (segment số 11). Rõ ràng là với cách đã nêu ta có thể ghi địa chỉ của 65535x65535 = 4 294 836 225 ô nhớ. Nói khác đi có thể tạo ra và ghi địa chỉ những bộ nhớ với dung lượng 4096 KB = 4 MB.

1. Bộ nhớ trong

Bộ nhớ trong bao gồm hai phần:

a. ***Bộ nhớ truy nhập trực tiếp (Random Access Memory)*** viết tắt là **RAM**. Bộ nhớ RAM được dùng để lưu trữ dữ liệu nhập vào từ bàn phím hoặc gọi ra từ bộ nhớ ngoài, lưu giữ các chương trình mà DOS nạp vào khi khởi động máy. Có thể ví nó giống như một kho trung gian chứa dữ liệu và khi mất điện thì dữ liệu cũng mất theo. Các máy tính thế hệ đầu được thiết kế với bộ nhớ RAM 256 K (1 K = 1024 bytes). Những máy vi tính thế hệ sau ví dụ loại AT 386 đã được thiết kế với dung lượng RAM đạt tới 4 MB (4096 K) Một số máy AT-486 có RAM đạt tới 16 hoặc 32 MB. Còn ngày nay dung lượng của Ram đã đạt đến cỡ 1 Gigabyte (1024 MB).

Với máy RAM đạt 4 MB có thể cài đặt WINDOWS 3.1 kèm theo WINWORD 6.0 và EXCEL 5.0. Còn nếu muốn chạy WINDOWS 95 thì phải có RAM đạt 8 MB. Hiện nay hầu hết các máy đều dùng hệ điều hành Win XP và bộ nhớ ram tối thiểu 128 MB. Nếu có điều kiện chúng ta nên chạy máy với RAM 512 MB hoặc 1 GB.

Để hình dung khả năng chứa của bộ nhớ này ta có thể làm một phép tính sau đây: Một trang sách khổ A4 (21x29,7 cm) có thể viết được nhiều nhất là 66 dòng, mỗi dòng có thể viết được 75 ký tự. Như vậy trên một trang có thể viết được 66x75 = 4950 ký tự. Với dung lượng nhớ 4096 KB tức là 4 194 304 bytes ta có thể đưa vào bộ nhớ của máy 847 trang sách. Nếu tính toán chi tiết hơn nghĩa là trừ đi các khoảng trống của lề trái, lề phải, lề trên, lề dưới... thì tổng số trang giấy có thể đưa vào bộ nhớ lên tới 1000 trang. Bộ nhớ RAM thường được chia một cách hình thức thành nhiều phần với tên gọi khác nhau ví dụ:

. Bộ nhớ quy ước: thường là 640 KB

. Bộ nhớ mở rộng: Extended memory

. Bộ nhớ bành trướng: Expanded memory

Ngoài ra tuỳ thuộc vào địa chỉ ô nhớ người ta còn phân thành vùng nhớ mé cao HMA (Hight Memory Area) hay vùng nhớ mé trên UMA (Upper Memory Area)

b. ***Bộ nhớ bất khả biến hay còn gọi là bộ nhớ chỉ đọc ROM (Read only Memory).***

Trong bộ nhớ ROM người sản xuất phần cứng ghi vào đó một số chương trình dùng để khởi tạo máy, để quản lý hoạt đông khi máy bắt đầu được đưa vào vận hành. Bộ nhớ ROM được nuôi bằng một nguồn điện acquy lắp ngay trên bảng chủ (Main Board hoặc Mother Board) do đó dữ liệu luôn luôn được bảo trì kể cả khi mất nguồn điện lưới. Những chương trình ghi trong ROM là độc quyền của người sản xuất do đó chúng ta không có quyền và cũng không có cách nào thay đổi được. Phần quản lý quan trọng nhất trong ROM là quản lý nhập-xuất dữ liệu BIOS (Basic Input/Output System) do đó trong quá trình Setup bộ nhớ ROM luôn được khai báo là ROM BIOS.

Ngoài Rom Bios bộ nhớ Rom còn các phần như Rom khởi động, Rom mở rộng và Rom Basic.

2. Bộ nhớ ngoài

Trong các máy tính PC hiện nay bộ nhớ ngoài bao gồm nhiều loại loại: Đĩa mềm (Flopy Disk), đĩa cứng (Hard Disk), đĩa CD-ROM, (Compact Disc Read-Only Memory), DVD và thẻ nhớ USB. Trước đây các máy tính thường được thiết kế với hai ổ đĩa mềm. 1,2 và 1,44 MB

Hiện nay các máy PC chỉ còn 1 ổ đĩa mềm 1.44 MB, 1 hoặc 2 đĩa cứng và 2 hoặc 4 cổng USB.

Mặc dù cấu trúc vật lý của đĩa hết sức phức tạp song nguyên lý hoạt động của nó lại rất đơn giản. Hai loại đĩa cứng và mềm đều dựa vào hiện tượng từ hoá để ghi dữ liệu. Đĩa được lắp vào ổ và quay với một tốc độ ổn định, tốc độ này khoảng từ 2400 - 4000 vòng/phút. Đầu từ được điều khiển áp vào cách bề mặt đĩa một khoảng nhỏ hơn 0,1 ηm. Khi đĩa quay các phần tử cực nhỏ trên bề mặt đĩa lướt qua đầu từ sẽ bị từ hoá và đảo hướng từ thông, trạng thái này tương ứng với số nhị phân 1. Những phần tử không bị từ hoá sẽ tương ứng với số nhị phân 0. Tập hợp các số nhị phân này sẽ cho ta hình ảnh một từ hoặc một ký tự của bảng mã quốc gia.

Các loại đĩa hiện đang dùng đều có hai mặt (**Side**) và được đánh số là 0 và 1. Trên từng mặt người ta chia làm nhiều vòng tròn đồng tâm, mỗi vòng tròn đó được gọi là một rãnh (**Track**). Track ngoài cùng (có đường kính lớn nhất) được đánh số là 0.. Đĩa 1,2 MB có 80 Track đánh số từ 0 đến 79. Tập hợp hai track đối xứng ở hai Side được gọi là **Cylinder**. Trên mỗi Track người ta lại phân thành một số phần đều nhau gọi là **Sector**,

trên đĩa 1,2 KB mỗi Track được chia thành 9 Sector, trên đĩa 1,44 MB có 18 Sector, còn trên đĩa cứng số Sector tuỳ thuộc vào loại đĩa. Dung lượng của mỗi Sector tuỳ thuộc vào loại đĩa song đều là bội số của 8 ví dụ: 128, 256, 512 và 1024. Đối với hệ điều hành MS-DOS kích thước Sector được chọn thống nhất là 512 bytes. Trên mỗi đĩa mềm Sector đầu tiên của Track số 0 thuộc Side 0 được đặt tên là **Boot Sector.** Đối với đĩa cứng người ta quy ước gọi Sector đầu tiên của side 1 (chứ không phải là side 0) là **Boot Master.**

Tất cả mọi loại đĩa trước khi đưa vào sử dụng đều phải tiến hành Format tức là ấn định cho đĩa số Track, số Sector trên mỗi Track, kích thước Sector ...Nếu lệnh Format có đi kèm theo tham số hệ thống /S (System) thì quá trình Format sẽ còn thực hiện việc chép vào đĩa hai File hệ thống là IO.SYS và MSDOS.SYS và tệp thông dịch COMMAND.COM.

Như đã nói dữ liệu được lưu trữ trên đĩa thông qua việc từ hoá các phần tử phủ trên bề mặt đĩa vì vậy bất kỳ một sự thay đổi cơ lý tính nào của đĩa cũng dẫn tới khả năng mất dữ liệu, đặc biệt là các vết xước trên bề mặt đĩa hoặc các hạt bụi có kích thước cực nhỏ chèn vào các Track. Khi cần mang đĩa đi xa tốt nhất là phải có hộp đựng cứng, tránh va chạm và cách ly những nơi có từ trường mạnh.

III. CÁC THIẾT BỊ VÀO-RA (INPUT-OUTPUT DEVICES)

Các ***thiết bị vào*** dùng để cung cấp dữ liệu cho máy xử lý, thiết bị vào thông dụng nhất hiện nay là bàn phím (Keyboard hay còn có tên là Console) và chuột (Mouse). Ngoài ra hiện nay người ta còn dùng thiết bị quét ảnh (Scaner) hoặc tiếng nói để cung cấp dữ liệu cho máy. ***Thiết bị ra*** dùng để đưa các kết quả đã xử lý cho người sử dụng, thiết bị ra thông dụng nhất hiện nay là màn hình (Monitor - Screen) và máy in (Printer). Máy in hiện nay có rất nhiều chủng loại ví dụ: máy in 9 kim, 24 kim, máy in Laser, máy vẽ Ploter.

Một trong những thiết bị đầu vào mới xuất hiện gần đây là máy quét Scaner. Các máy PC hiện đại có thể tích hợp thêm các bộ phận thu nhận tín hiệu Tivi, tín hiệu Internet không dây... Nếu được trang bị thêm thiết bị quét ảnh SCANER thì có thể làm rất nhiều công việc về đồ hoạ. Thiết bị đầu ra chính là màn hình màu với độ phân giải cao (1024x768) sẽ cho ta chất lượng hình ảnh sát với thực tế. Với chế độ làm việc WYSIWYG (What you see is what you get - Cái gì nhìn thấy trên màn hình thì sẽ in ra trên giấy đúng như vậy) và một máy in LASER chúng ta sẽ có được những trang in hoàn hảo mà không một thiết bị in nào khác có thể sánh được.

1. Bàn phím

Thuật ngữ thông dụng để chỉ bàn phím là Keyboard, tuy nhiên một số tài liệu lại dùng là Console. Bàn phím được chia thành 5 khu vực phím:

. Các phím chức năng ký hiệu từ F1 đến F12, ngoài ra còn một số phím khác bố trí giải rác như: Print Screen, Scroll Lock, Pause, Insert, Caps Lock, Num Lock,

. Các phím ký tự a...z, 1..0, +,-,.,/...

. Các phím chức năng phụ Alt, Ctrl, Shift....

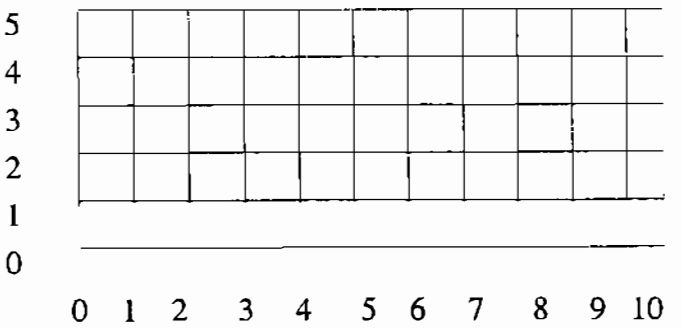
. Các phím dịch chuyển Home, End, PageUp, PageDown, Space....

. Nhóm phím số và toán tử kích hoạt bằng phím Numlock

Nguyên lý làm việc của bàn phím như sau:

Trên bàn phím có các mạch điện nằm ngang và nằm dọc không tiếp xúc với nhau (hình 2.3).

Khi ta bấm một phím tức là làm chập mạch tại một vị trí nào đó, việc này tạo nên một xung điện gọi là mã quét (Scan Code). Mã quét này được CPU xử lý thông qua Ngắt quản lý bàn phím số 9 (Int 9). Ngắt số 9 sẽ điều khiển chương trình có trong ROM-BIOS để thực hiện các công việc:



Hình 2.3

- Chuyển mã quét sang mã ASCII
- Đưa mã quét và mã ASCII vào bộ đệm bàn phím
- Chọn ký tự từ bộ đệm bàn phím đưa ra màn hình (hoặc máy in).

2. Màn hình

Màn hình là một khối thuỷ tinh có dạng hình phễu chân không bên trong có một thiết bị phóng tia điện tử. Mạt trong của màn hình phía đối diện với ống phóng điện tử được phủ một lớp huỳnh quang. Lớp huỳnh quang này sẽ phát sáng khi các điện tử đập vào với một cường độ đủ lớn. Màn hình của các máy vi tính hiện nay được thiết kế với nhiều chủng loại khác nhau:

. Màn hình Monochrom, Hercule là các màn hình đơn sắc

. Màn hình Ega, Cga, Vga, Tvga, Svga là các màn hình màu

Màn hình của các máy tính cá nhân thông thường đều có chiều dài đường chéo 14 Inch và được thiết kế để có thể làm việc ở hai chế độ: chế độ văn bản (Text) và chế độ đồ hoạ (Graphic). Chất lượng của một màn hình thể hiện qua hai thông số cơ bản:

a. Độ phân giải: (Resolution) là số điểm sáng mà màn hình có thể hiển thị. Độ phân giải ký hiệu bởi hai chữ số, chữ số đầu thể hiện số điểm trên một dòng, số sau là số điểm trên một cột. Màn hình VGA độ phân giải là 640x480, màn hình SVGA 1024x720, Màn hình Hercule 720x340.... Với các máy tính hiện đại độ phân giải màn hình có thể thay đổi khi chọn chức năng Settings trong Control Panel.

b. Màu sắc: Tuỳ thuộc vào bộ nhớ màn hình mà chúng ta có số lượng màu có thể lựa chọn, Màn hình VGA cho ta 16 màu, màn hình Sgva cho 256 màu.

IV. KHỐI XỬ LÝ TRUNG TÂM (CENTRAL PROCESSING UNIT)

Khối xử lý trung tâm là bộ não của máy vi tính. Nhiệm vụ của khối xử lý là thực hiện các phép tính toán số học và lôgic, đồng thời quyết định các thao tác mà chương trình đòi hỏi. Khối vi xử lý là các vi mạch bán dẫn được lắp ráp trong một Block kín (chúng ta quen gọi là IC), các thế hệ cũ từ 8085 đến 80286 IC có 64 chân, đây là vi xử lý 16 bit, thế hệ thông dụng hiện nay và 486 và 586 và khả năng xử lý lên tới 64 bit.

Xét về phương diện thông tin một bộ vi xử lý phải luôn luôn thực hiện 3 nhiệm vụ sau đây:

- Nhập vào hoặc xuất ra các số liệu của chương trình

- Thực hiện việc xử lý các số liệu đưa vào

- Quản lý và cất giữ các số liệu vào các thiết bị nhớ trong hoặc ngoài

Để thực hiện được các nhiệm vụ đó khối vi xử lý được thiết kế gồm các bộ phận:

1. Bộ điều khiển

Bộ điều khiển bao gồm một đồng hồ điều khiển nhịp đồng bộ và một số thanh ghi, ví dụ thanh ghi đếm lệnh PC (Program Counter), thanh ghi lệnh IR (Instruction Register), thanh ghi trạng thái ST (Status). Giá tị trong thanh ghi PC chính là số hiệu ô nhớ chứa lệnh sắp thực hiện, khi lệnh được thực hiện xong giá trị trong PC sẽ thay đổi để chỉ đến lệnh tiếp theo. Thanh ghi IR dùng để ghi các lệnh đang được giải mã thực hiện còn thanh ghi ST dùng để ghi các trạng thái các bộ phận cần quản lý trong đó bao gồm cả trạng thái của việc thực hiện các phép toán như trạng thái nhớ (tức là phép nhớ lên bit cao hơn), trạng thái tràn ô nhớ (phép chia cho số 0) v.v...

2. Bộ xử lý

Bộ xử lý thường được thiết kế đồng bộ vừa xử lý số học vừa xử lý lôgic (Arithmetic Logic Unit - ALU) . Bộ xử lý làm nhiệm vụ tính toán các phép toán số học

và lôgic trên các biến đã khai báo. Trước hết nó phân loại phép toán thuộc nhóm nào (số học hay lôgic) sau đó thực hiện các tính toán và ghi nhớ trạng thái của kết quả vào các thanh ghi.

3. Các thanh ghi

Các thanh ghi là những ô nhớ đặc biệt, loại thanh ghi 32 bit hiện nay được chia thành các nhóm sau :

- Thanh ghi đa năng AX, BX, CX,DX
- Thanh ghi đoạn CS, BS, DS, ES, GS, SS
- Thanh ghi offset IP, DI, BP, SP, GP, SI
- Thanh ghi điều khiển RC0, RC1, RC2, RC3, RC6, RC7
- Thanh ghi kiểm tra TR0, TR1 ..... TR7
- Thanh ghi cờ (FLAG) F
- Thanh ghi Stack dùng để lưu giữ và phục hồi trạng thái làm việc mỗi khi có lệnh xin ngắt quá trình xử lý để tạm thời làm công việc khác.

Các thanh ghi 32 bit được chia thành hai nửa , nửa dưới từ bit 0 đến bit 15 là vùng có thể thâm nhập vào được, nghĩa là ta có thể thay đổi các giá trị của 16 bit này thông qua lệnh R của DEBUG. Nửa trên từ bit 16 đến bit 32 là vùng kín thuộc quyền quản lý của ROM BIOS.

4. Các BUS dữ liệu

Đây là hệ thống các đường truyền dẫn bảo đảm việc truyền dữ liệu giữa các bộ phận bên trong vi xử lý và từ vi xử lý với bên ngoài.

Phần mềm là các chương trình có thể chạy trên máy vi tính, mỗi phần mềm chỉ có thể chạy trên một số loại máy nhất định. Trước khi cài đặt các phần mềm vào máy cần kiểm tra xem chúng có bị nhiễm virus hay không và đặc biệt lưu ý các phần mềm đó có cho phép tự do cài đặt hay không.

Với các máy PC thế hệ mới thiết bị đầu ra trở nên rất phong phú, kết hợp một ổ DVD với một Projector chúng ta có thể xem các phim ghi trên đĩa.

V. VIRUS TIN HỌC

Ngày nay không mấy ai còn ngạc nhiên khi phát hiện thấy máy tính của mình bị nhiễm Virus song điều đáng nói là cần phải hiểu thế nào cho đúng khái niệm Virus tin học. Trước hết cần phải nói rằng không phải bất kỳ một lỗi nào của máy tính dẫn tới mất mát dữ liệu hay ngừng các quá trình xử lý đều là sản phẩm của Virus. Ngay cả khi đã phát hiện Virus trên máy của bạn thì cũng đừng quá lo lắng mà vội vã Format lại đĩa.

Virus tin học thực chất là một đoạn chương trình có kích thước cực kỳ nhỏ bé nhưng lại bao hàm trong nó những chức năng rất đa dạng. Vì lẽ đó Virus được thiết kế trực tiếp bằng ngôn ngữ Assembler. Trong khi đối với người xử dụng Virus thực sự là một thảm hoạ thì đối với người lập trình về một khía cạnh nào đó cấu trúc của một Virus lại là một công trình đáng tự hào. Chúng ta có thể sốt ruột với một câu hỏi:" Virus xuất hiện từ đâu và vào lúc nào ". Đáng tiếc là không ai có thể trả lời chúng ta một cách chính xác được. Một số tài liệu cho rằng Virus xuất hiện lần đầu tiên vào năm 1987 trong khi đó bản thân Virus Brain khi kết thúc sự phá hoại của mình lại cho hiện lên trên màn hình dòng thông báo năm sinh của nó là năm 1986. Tuy mới chỉ có lịch sử phát triển khoảng hơn một chục năm song họ hàng nhà Virus ngày nay cũng đã lên đến con số hàng vạn đấy là chưa kể những Virus còn đang tiềm ẩn chưa xuất đầu lộ diện.

Virus xuất hiện không phải là kết quả của một sự ngẫu nhiên mà là những công trình được nghiên cứu cẩn thận. Để bảo vệ bản quyền chống mọi hành động sao chép trộm, các nhà sản xuất phần mềm đã đưa ra ý tưởng dấu vào trong chương trình một đoạn mã phá hoại. Khi chương trình bị sao chép thì đoạn mã này sẽ hoạt động gây nên tổn thất không thể lường trước được cho kẻ ăn trộm. Ngoài các nhà kinh doanh phần mềm thì các trường đại học cũng là cái nôi sản sinh ra Virus, bởi lẽ nhiều sinh viên trẻ có tài năng, được đào tạo nghiêm túc nhưng lại chưa có đủ sự chín chắn cần thiết, những người này xem việc tạo được một Virus là một niềm tự hào mà không hiểu rằng để diệt nó người ta phải tốn không ít mồ hôi và tiền của.

Căn cứ vào tính chất của đoạn mã phá hoại người ta phân chúng thành hai loại:

1. Trojan Horse: (Con ngựa thành Troa)

Thuật ngữ này bắt nguồn từ truyện thơ của Homer trong thần thoại Hylạp. Các đoạn mã phá hoại kiểu này hoàn toàn không có tính chất lây lan, nó được gài vào một phần mềm nào đó và được nguỵ trang kín đáo. Đến một thời điểm do tác giả định trước đoạn mã này sẽ hoạt động như một chương trình độc lập và đối tượng phá hoại là kho dữ liệu của đối thủ tức là đĩa cứng hoặc đĩa mềm. Nếu đoạn mã này chứa lệnh Format đĩa thì hậu quả của nó không thể lường hết được.

2. Virus tin học

Thuật ngữ này dùng để chỉ những chương trình máy tính có khả năng tự sao chép nó lên những đĩa hoặc File khác mà người sử dụng không hay biết. Do tầm vóc của nó vĩ đại hơn Trojan Horse nên hậu quả phá hoại của nó cũng ghê gớm hơn, mang tính toàn cầu hơn. Tuỳ theo tính chất lây lan và mục tiêu phá hoại người ta chia Virus thành hai loại: B-virus: Là những Virus chỉ tấn công lên những Boot Sector (đối với ổ mềm) hoặc Mater Boot (đối với ổ cứng) F-Virus: Là những Virus chỉ tấn công vào các file thi hành được, hiểu một cách thông thường đó là các File có đuôi .COM hoặc .EXE . Tuy vậy ở nước ta cũng đã xuất hiện những loại Virus chỉ tấn công vào các tệp dữ liệu, đó là các

tệp có đuôi DOC (của Winword) hoặc có đuôi .DBF tức là các tệp dữ liệu sử dụng trong FOXBASE hoặc FOXPRO.

Để phòng chống Virus cách tốt nhất là thường xuyên kiểm tra đĩa. Đặc biệt thận trọng khi sao chép các phần mềm từ nơi khác mang tới. Trên thế giới phần mềm diệt Virus có khá nhiều song mỗi phầm mềm chỉ diệt được một số loại Virus nhất định. Các tác giả trong nước cũng đã bỏ ra nhiều công sức xây dựng các phần mềm chống Virus, hiện có nhiều chương trình đang được lưu hành như ATV, BKAV, D2, ...

Nói chung nên sử dụng các chương trình Scan (dò tìm) hàng ngày, khi phát hiện Virus thì dùng chương trình diệt chúng, nên dùng kết hợp các phần mềm diệt Virus của nước ngoài với các chương trình do các tác giả Việt Nam lập. Sau khi đã chạy các chương trình diệt Virus cần tắt máy và rút nguồn điện khoảng 5 phút hãy khởi động lại.

Các đơn vị sử dụng máy cần bảo quản đĩa ngay tại phòng máy của đơn vị mình, tránh đem đĩa đi dùng ở nơi khác. Đặc biệt lưu ý không để đĩa gần những nơi có từ trường mạnh, có nhiệt độ quá cao hoặc quá thấp. Mọi tác động về cơ học, hoá học đều có thể làm hỏng đĩa. Trong trường hợp đĩa có sự cố (đặc biệt là đĩa cứng) cần phải mời các chuyên gia am hiểu đến khắc phục, không nên mò mẫm làm dễ dẫn tới mất toàn bộ dữ liệu.

Chương 3
THUẬT GIẢI VÀ LƯU ĐỒ

I. KHÁI NIỆM VÀ TÍNH CHẤT CỦA THUẬT GIẢI

1. Khái niệm chung

Thuật giải (hay còn gọi là thuật toán) là một dãy tuần tự các bước xử lý để giải quyết một bài toán cho đến kết quả cuối cùng, hoặc các kết quả trung gian phục vụ cho một tiến trình xử lý khác. Thuật giải là một trong những lĩnh vực nghiên cứu quan trọng nhất của Tin học hiện đại. Không phải mọi thuật giải đều đưa đến kết quả cuối cùng với độ chính xác mong muốn.

Xét một số ví dụ sau:

Ví dụ 1:

Có n vật thể và một chiếc cân đĩa không có quả cân. Hãy xác định vật thể nặng nhất.

Phương pháp tiến hành là đặt hai vật thể lên đĩa cân, bỏ vật thể nhẹ ra chỗ khác , giữ vật nặng trên đĩa cân. Lấy tiếp một vật thể khác đặt lên đĩa cân và lại bỏ vật nhẹ đi, lặp lại quá trình trên cho đến khi hết vật thể . Vật thể cuối cùng còn lại trên đĩa cân là vật thể nặng nhất.

Phương pháp trên được mô tả theo các bước như sau:

1. Đặt hai vật thể lên hai đĩa cân

2. Bỏ đi vật thể nhẹ (vật thể này sẽ không tham gia vào quá trình cân nữa)

3. Kiểm tra xem còn vật thể hay không

* Nếu còn thì đặt tiếp 1 vật thể lên đĩa cân trống, quay lại thực hiện bước 2.

* Nếu không thì thực hiện bước 4

4. Kết thúc công việc , vật còn lại là nặng nhất

Ví dụ 2:

Hai người A và B tham gia trò chơi bốc sỏi. Có 37 viên sỏi và nguyên tắc bốc như sau: Hai người bốc lần lượt, mỗi lần bốc ít nhất là 1 viên và nhiều nhất 5

viên. Ai là người bốc cuối cùng sẽ thua cuộc. Hãy tìm thuật giải để người bốc sau luôn luôn thắng cuộc.

Giả thiết rằng A bốc trước B bốc sau, muốn thắng cuộc, lần bốc cuối cùng B phải chừa lại 1 viên sỏi để A phải bốc. Như vậy cả hai người trong những lần trước đó đã bốc hết 36 viên. Dễ dàng nhận ra rằng nếu mỗi đợt cả hai bốc 6 viên thì sau 6 lần bốc sẽ chừa lại 1 viên. Đây chính là bí quyết để B luôn thắng cuộc. Căn cứ vào số viên sỏi mà A đã bốc, B sẽ bốc số viên sao cho cộng với số viên A đã bốc là bằng 6.

Thuật giải của bài toán sẽ gồm các bước sau:

Giả thiết rằng theo thứ tự A bốc trước , B bốc sau.

1. B ghi nhớ số viên sỏi mà A đã bốc (giả sử là n)

2. B bốc số viên sỏi bằng 6-n

3. Nếu số viên sỏi còn nhiều hơn 1 thì lặp lại bước 1, nếu chỉ còn lại 1 viên thì chắc chắn đến lượt A phải bốc.

4. Kết thúc cuộc thi

2. Tính chất của thuật giải

Qua hai ví dụ trên ta có thể nhận thấy thuật giải có những tính chất đặc trưng đó là tính tuần tự, tính kết thúc, tính chính xác, tính hiệu quả và tính phổ cập.

a. Tính tuần tự

Các bước của thuật giải được thực hiện lần lượt từ trên xuống dưới việc đảo ngược các bước không cho ta kết quả hoặc sẽ dẫn tới kết quả sai. Trong cả hai ví dụ trên việc đảo ngược các bước ví dụ giữa bước 1 và 2 sẽ làm cho thuật giải trở nên vô nghĩa, một số trường hợp khác việc đảo ngược vẫn làm cho thuật giải có nghĩa song kết quả của thuật giải lại không đúng nữa.

b. Tính kết thúc

Một thuật giải phải bảo đảm tính kết thúc sau một số hữu hạn các bước xử lý. Số bước có thể là rất nhiều, thời gian xử lý có thể là rất lâu song phải là hữu hạn. Trong một số trường hợp số bước xử lý của thuật giải rất ít song không kết thúc như ví dụ sau đây sẽ dẫn tới tình trạng là máy tính sẽ làm việc chừng nào nó chưa bị hỏng hoặc chưa bị cắt điện.

Ví dụ: Thuật giải quay vòng vô hạn

1. Đọc số n vào bộ nhớ

2. Tính căn bậc hai của n, hiện kết quả lên màn hình

3. Quay trở lại bước 1

Với thuật giải trên đây rõ ràng là chúng ta có được kết quả tính căn bậc hai của một số song chúng ta không có cách nào kết thúc được chương trình trừ phi phải tắt máy.

c. Tính chính xác

Thuật giải lập ra phải bảo đảm tính chính xác của kết quả. Sử dụng cùng một thuật giải trên hai bộ xử lý khác nhau phải cho kết quả với độ chính xác như nhau.

d. Tính hiệu quả

Tính hiệu quả của một thuật giải thể hiện ở chỗ nó bảo đảm sai số theo yêu cầu của các kết quả tính toán song thời gian xử lý là ngắn nhất, sử dụng ít tài nguyên nhất. Cùng một bài toán có nhiều cách xây dựng thuật giải, người lập trình giỏi là người biết lựa chọn thuật giải nào mang lại hiệu quả cao nhất.

e. Tính phổ cập

Tính phổ cập thể hiện ở chỗ một thuật giải có thể áp dụng cho nhiều bài toán đồng dạng. Với thuật giải trong ví dụ 2 chúng ta có thể mở rộng cho trường hợp 55 viên sỏi, mỗi lần bốc nhiều nhất 6 viên và ít nhất 3 viên.....

Để lập trình giải quyết bài toán trong một ngôn ngữ nào đó chúng ta phải phân tích thuật giải thành những đoạn thuật giải nhỏ hơn, cứ thế cho đến khi không thể phân nhỏ được nữa. Những phần nhỏ nhất của thuật giải sẽ tương ứng với một cấu trúc lệnh nào đó đã được thiết kế trong ngôn ngữ mà ta lựa chọn để lập trình. Nói khác đi đó chính là các lệnh mà ta cần ứng dụng để giải quyết bài toán.

II. LƯU ĐỒ (SƠ ĐỒ KHỐI)

Để mô tả một thuật giải người ta sử dụng ngôn ngữ thuật giải, đó hoặc là các lệnh của một ngôn ngữ nào đó viết dưới dạng văn bản hoặc là một lưu đồ. Lưu đồ trình bày một chương trình dưới dạng các khối, các mũi tên chỉ hướng. Có 3 loại khối cơ bản sau:

1. Khối các đoạn chương trình (ký hiệu bởi hình chữ nhật)

2. Khối chuyển nhánh (ký hiệu bởi hình thoi)

3. Khối lặp lại một đoạn chương trình (ký hiệu bởi mũi tên vòng lặp bao quanh các khối khác).

4. Khối thể hiện sự bắt đầu hoặc kết thúc (hình chữ nhật lượn tròn hai đầu).

Ngoài các khối, trong ngôn ngữ thuật giải còn sử dụng một khái niệm gọi là phép gán. Phép gán ký hiệu là ":= ".

Phép gán là việc chuyển giá trị (viết ở bên phải dấu gán) cho một đại lượng đã được định nghĩa trước (viết ở bên trái dấu gán).

Ví dụ: " Ngày " được định nghĩa là đại lượng có trị số nguyên từ 1 đến 31. Để gán cho Ngày giá trị 15 ta viết

Ngày:= 15

Hoặc Xedap là đại lượng chỉ nhận các giá trị là chữ viết trong tập hợp sau [Viha, Thongnhat, Mifa, Hoankiem, Sk, Favorit].

Ta có phép gán

Xedap:= 'Thongnhat'; Xedap:= 'Mifa' ; ...

1. Bắt đầu và kết thúc chương trình

Để thể hiện vị trí bắt đầu và kết thúc một chương trình chúng ta dùng hình chữ nhật với hai phần bán nguyệt ở hai đầu (hình 3.1a).

2. Khối xử lý

Đây là khối các lệnh xử lý một công việc nào đó bao gồm việc nhập dữ liệu,tính toán, viết kết quả ... khối xử lý đặt trong một hình chữ nhật (hình 3.1b).

a/ b/

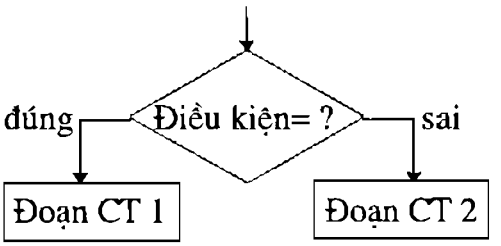
Hình 3.1

3. Khối chuyển nhánh

Căn cứ vào những điều kiện nhất định chương trình sẽ chuyển nhánh sang một đoạn khác nếu điều kiện thoả mãn. Có hai loại khối chuyển nhánh:

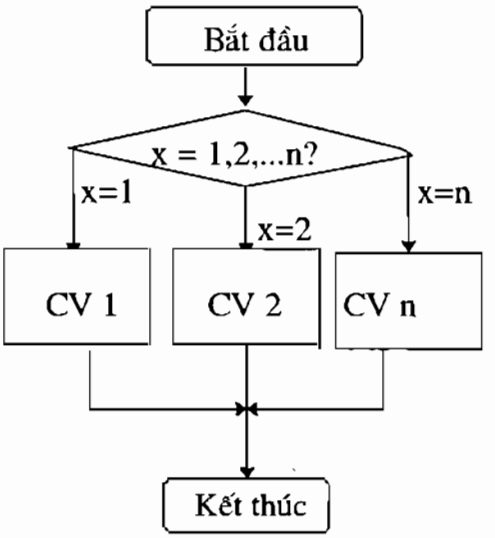
a. Căn cứ vào điều kiện đúng hay sai của hàm lôgic chương trình sẽ chuyển sang thực hiện hoặc công việc 1 hoặc công việc 2 (hình 3.2).

b. Căn cứ vào giá trị của một biến cho trước chương trình sẽ thực hiện công việc tương ứng (hình 3.3).



Hình 3.2

Kiểm tra điều kiện
Nếu điều kiện = đúng
  Đoạn CT 1
Nếu không
  Đoạn CT 2
Hết nếu



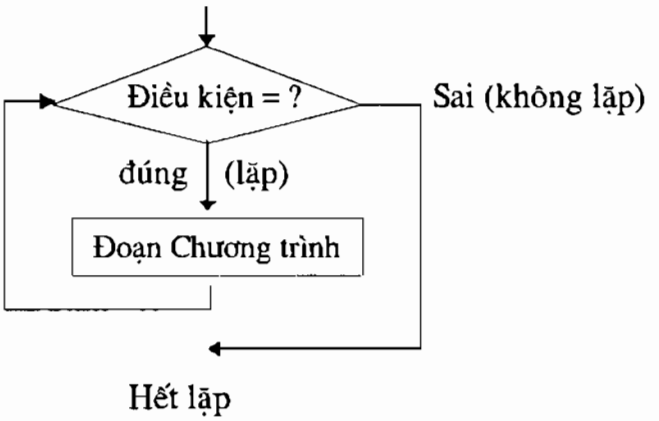
Kiểm tra trường hợp
x = 1,2,3,.....n
Trường hợp x=1
Công việc 1
x=2 Công việc 2
.....
x=n Công việc n
Hết trường hợp

Hình 3.3. Khối chuyển sang 1 trong nhiều trường hợp

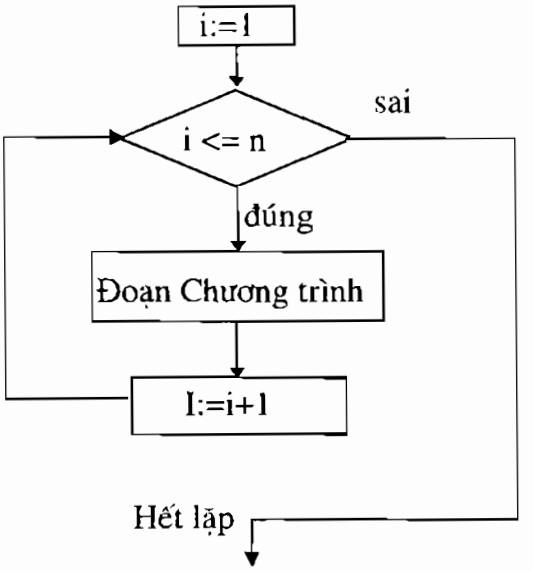
4. Khối chỉ định lặp

Sơ đồ khối chỉ định lặp chỉ ra như trong hình 3.4. Quá trình lặp thực hiện như sau: Nếu điều kiện còn đúng thì thực hiện đoạn chương trình, nếu không thì ra khỏi cấu trúc lặp thực hiện tiếp các lệnh của chương trình sau cấu trúc lặp.

Còn một cấu trúc lặp khác với số bước lặp xác định, ở đây không có việc kiểm tra điều kiện mà chỉ kiểm tra số lần đã lặp (hình 3.5).

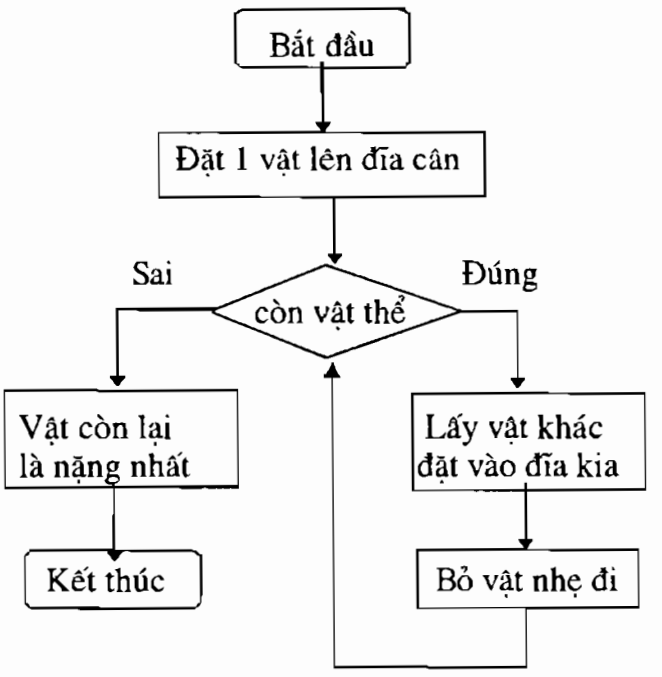


Hình 3.4 : Sơ đồ khối chỉ định lặp



Hình 3.5 Sơ đồ khối lặp n lần

Ứng dụng các hình vẽ đã quy ước chúng ta có thể vẽ lưu đồ bài toán tìm vật thể nặng nhất trong n vật thể đã nêu trên (hình 3.6).



Hình 3.6. Lưu đồ bài toán tìm thể nặng nhất

Ví dụ 4: Thuật giải và lưu đồ bài toán giải phương trình bậc hai

$ax^2 + bx + c = 0$ với a # 0

Thuật giải bài toán này đã được biết trong chương trình phổ thông.

Bắt đầu

1. Nhập các giá trị a,b,c
2. Tính biệt thức D của phương trình

   $D = b^2 - 4ac$
3. Xét dấu của D trong 2 trường hợp

   * Nếu D < 0 --> không có nghiệm thực

   Còn nếu

   * D >= 0 --> $x_1 = (-b + D) / 2a$;

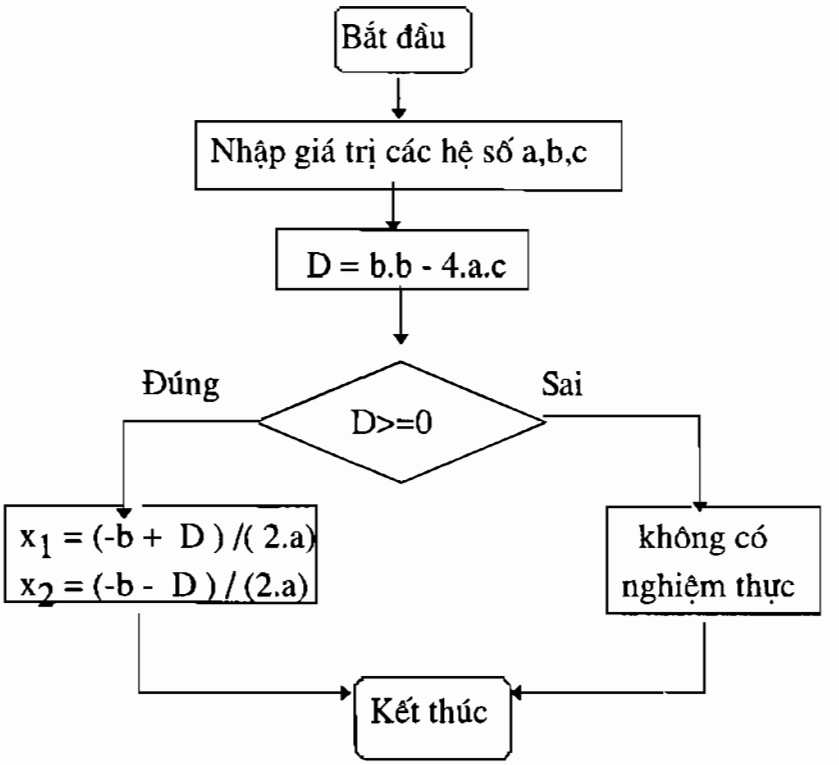
   $x_2 = (-b - D) / 2a$

   * Hết nếu

4. Ghi kết quả ra thiết bị in.

Kết thúc.

Lưu đồ của bài toán giải phương trình bậc hai trình bày trên hình 3.7



Hình 3.7

III. MỘT SỐ VÍ DỤ VỀ THUẬT GIẢI

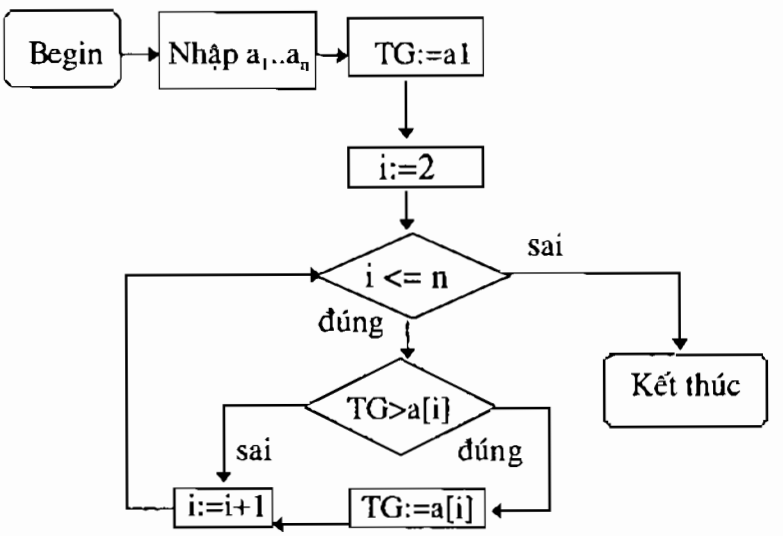
1. Thuật giải tìm số lớn nhất hoặc bé nhất trong một dãy số

Dãy số a_1, a_2, a_3,.....a_n được xem là một mảng n phần tử và được ký hiệu là a[i] với i = 1,2...n . Thuật toán tìm số lớn nhất trong mảng có một số khác biệt so với thuật toán tìm vật thể nặng nhất đã trình bày ở trên, ở đây ta sử dụng một đại lượng trung gian là TG, và một biến đếm là i.

Các bước của thuật giải:

1. Nhập vào mảng a[i] giá trị của các phần tử a[1], a[2],....
2. Gán giá trị a[1] cho TG và gán số 2 cho i
3. So sánh TG với a[i]
4. Lấy số lớn hơn trong hai số gán trở lại cho TG
5. Tăng giá trị i thêm 1 đơn vị
6. Quay về bước 3 cho đến khi i = n thì sang bước 7
7. Giá trị cuối cùng có trong TG là số lớn nhất của dãy số
8. Ghi kết quả ra thiết bị in
9. Dừng chương trình

Lưu đồ



Hình 3.8 Lưu đồ bài toán tìm số lớn nhất

2. Tìm trong dãy số $a_1, a_2, ... a_n$ một số có giá trị bằng k

Nhận xét rằng việc tìm kiếm sẽ dẫn đến 1 trong hai khả năng:

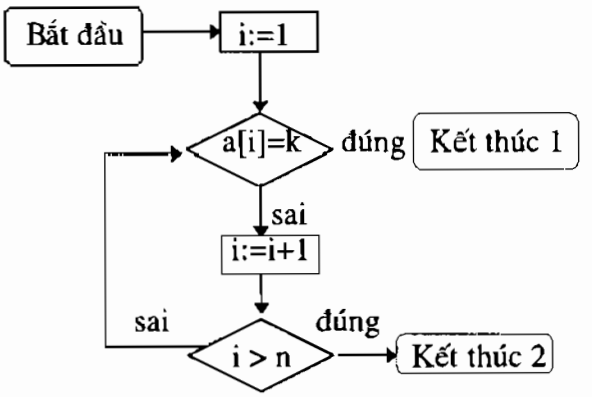
- Tìm thấy một số a[i] = k với 1<= i <= n
- Không tìm thấy một số nào bằng k

Trong cả hai trường hợp dù tìm thấy hay không tìm thấy cũng vẫn phải kết thúc tìm kiếm, nghĩa là thuật toán phải có hai khả năng kết thúc.

Thuật giải

1. So sánh số thứ nhất với k
2. Nếu1
   - bằng nhau: Kết thúc 1
   - không bằng nhau
3. Kiểm tra xem còn số chưa so sánh hay không
   Nếu2
   - còn: so sánh số tiếp theo với k
     lặp lại bước hai
   - hết: Kết thúc 2

   Hết nếu 1
   Hết nếu 2
4. Dừng chương trình



Hình 3.9. Lưu đồ bài toán tìm một số bằng k trong dãy số $a_1 \ldots a_n$

3. Tính diện tích tam giác biết chiều dài ba cạnh a,b,c

Nhận xét : Vì không biết bất kỳ chiều cao nào của tam giác nên không thể dùng công thức quen thuộc: diện tích = (đáy x chiều cao)/2. Gọi nửa chu vi của

tam giác là p (p = (a+b+c)/2) thì diện tích của tam giác S có thể tính qua công thức:

$$S = \sqrt{p*(p-a)*(p-b)*(p-c)}$$

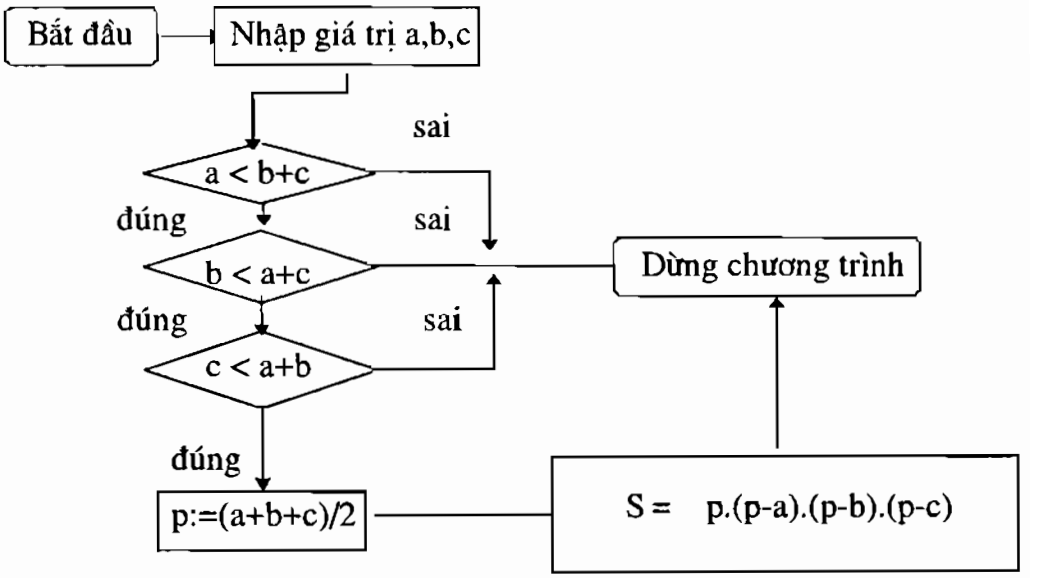
Ta có thuật giải bài toán

1. Nhập các giá trị a,b,c
2. Kiểm tra xem các kích thước a,b,c có tạo nên tam giác hay không?
3. Nếu

   không: kết thúc

   có : p := (a+b+c)/2 ; S := $\sqrt{p*(p-a)*(p-b)*(p-c)}$

   Hết nếu
4. Ghi kết quả ra thiết bị ghi, dừng chương trình



Hình 13: Lưu đồ bài toán tính diện tích tam giác

Chương 4

HỆ ĐIỀU HÀNH MÁY VI TÍNH

A. HỆ ĐIỀU HÀNH MS-DOS

I. KHÁI NIỆM HỆ ĐIỀU HÀNH - ĐĨA HỆ THỐNG

Hệ điều hành OS (Operating System) là công cụ giao tiếp giữa máy tính với các đối tượng ngoài máy tính. Hệ điều hành là phần mềm điều khiển toàn bộ tiến trình xảy ra trong máy tính. Hiện nay trên thế giới có nhiều hệ điều hành khác nhau đang được sử sụng song thông dụng nhất là hệ điều hành do hãng Microsoft soạn thảo, phiên bản mới nhất của hệ điều hành do hãng này đưa ra là WINDOWS XP. Hệ điều hành WINDOWS có thể chuyển về làm việc ở Mode DOS sẽ tương đương với Version 7.0 do vậy những thao tác của DOS vẫn áp dụng bình thường ngay cả khi cài đặt WINDOWS XP. Sở dĩ trong giáo trình này chúng ta vẫn còn đề cập đến hệ điều hành DOS là vì chúng vẫn còn được sử dụng khi lập trình với ngôn ngữ bậc thấp.

Vì là công cụ phần mềm hết sức quan trọng nên hệ điều hành luôn luôn được cải tiến, phiên bản cuối cùng của hệ điều hành DOS là version 7.0 . Hệ điều hành thực chất là một bộ chương trình hoạt động giống chức năng của một người phiên dịch giữa hai đối tượng là Người sử dụng và Máy tính. Thông qua DOS người sử dụng tạo lập hoặc xoá các tệp (File), các thư mục (Directory), chạy và ghép nối các chương trình, sao chép các tệp, chuyển thông tin tới các thiết bị ngoại vi v.v...

Trong trường hợp cần thiết chúng ta có thể tự tạo cho mình đĩa hệ thống bằng lệnh FORMAT với tham số /S kèm theo. Đĩa hệ thống đơn giản nhất bao gồm ba tệp là IO.SYS, MSDOS.SYS và COMMAND.COM. Hai tệp IO.SYS, MSDOS. SYS là tệp ẩn do đó khi gõ lệnh DIR ta chỉ thấy tệp COMMAND.COM

Với hệ điều hành MS-DOS sau khi đã FORMAT tạo đĩa hệ thống chúng ta còn cần tạo ra hai tệp nữa là AUTOEXEC.BAT và CONFIG.SYS, nội dung cụ thể của hai tệp này sẽ trình bày ở phần sau. Trên đĩa hệ thống còn có thể chép vào một số lệnh ngoại trú của DOS như:

FORMAT.COM, UNFORMAT.COM, SYS.COM, XCOPY.DEBUG.EXE BACKUP.EXE, DISKCOPY.COM, DRIVER. SYS, RAMDRIVE.SYS, FDISK.EXE, RESTORE.EXE DOSKEY.COM .....

Lưu ý: *Không phải bất cứ đĩa nào chép đủ các tệp trên cũng trở thành đĩa hệ thống, muốn có đĩa hệ thống trong quá trình format (tạo khuôn đĩa) nhất thiết phải đưa vào tham số /S, về điều này ta sẽ xét kỹ trong lệnh FORMAT ở phần sau.*

Hệ điều hành có thể cài đặt vào đĩa cứng hoặc đĩa mềm. Nếu là đĩa mềm thì phải lắp đĩa vào ổ trước khi khởi động máy, cần lưu ý rằng hệ thống chỉ có thể khởi động từ ổ A hoặc C do vậy nếu trên máy có hai ổ đĩa mềm là A và B thì không thể lắp đĩa hệ thống vào khởi động từ ổ B. Các máy sản xuất gần đây (AT-386, AT-486) thường có hai ổ đĩa mềm, một ổ có dung lượng 1,2 MB và ổ còn lại có dung lượng 1,44 MB, người sử dụng có thể chọn một trong hai ổ này làm ổ A bằng cách đảo vị trí đầu nối của BUS truyền dẫn bên trong thân máy và thực hiện khai báo dung lượng khi chạy chương trình SETUP. Trường hợp máy chỉ có một ổ dung lượng 1,44 Mb thì phải khai nó là ổ A để có thể khởi động từ ổ này.

II. MỘT SỐ QUY ĐỊNH CỦA HỆ ĐIỀU HÀNH MS-DOS

Các lệnh của DOS chia thành hai loại: lệnh nội trú và lệnh ngoại trú. Lệnh nội trú là các lệnh đã được đưa vào trong ROM-BIOS dưới dạng các FUNCTIONS và sẽ được chỉ đến thông qua các Vector ngắt. Sau khi máy khởi động máy nếu ta gõ tên lệnh từ dấu nhắc thì nó sẽ được thực hiện ngay tức khắc. Lệnh ngoại trú thực chất là các tệp chương trình được lưu trên đĩa, muốn thực hiện lệnh ngoại trú thì trước hết phải nạp chúng từ đĩa từ vào bộ nhớ. Trong thư mục DOS các tệp có đuôi .COM, .EXE đều được coi là lệnh ngoại trú. Cần phân biệt các lệnh của DOS với các chương trình được soạn thảo bằng các ngôn ngữ lập trình như Pascal, Turbo C++, Assemble... Các chương trình này khi dịch sang dạng mã máy cũng có đuôi là EXE hoặc COM song đó không phải là các lệnh của DOS. Tất cả các lệnh DOS được gõ sau dấu nhắc hệ thống chỉ được thực hiện sau khi ta gõ phím Enter. Trước khi nghiên cứu các lệnh của DOS cần chú ý một số quy định khi viết các câu lệnh.

1. Quy định ổ đĩa

DOS đặt tên các ổ đĩa bằng chữ cái bắt đầu bằng chữ A , nếu máy có hai ổ đĩa mềm thì DOS đặt tên là A và B. Nếu máy có thêm một ổ đĩa cứng thì DOS đặt tên là C v.v... Tên ổ đĩa trong các lệnh của DOS luôn luôn có dấu ": " bên cạnh, ví dụ A:, B:, E:

Lưu ý: Trong mỗi máy vi tính hiện nay đều có thiết kế vị trí để có thể lắp đặt 2 ổ đĩa cứng khi đó tên các ổ đĩa cứng sẽ là C và D. Các đĩa cứng lắp vào các ổ vật lý này được này được gọi là đĩa vật lý để phân biệt với đĩa Logic. Đĩa Logic là các đĩa giả, thực chất nó là một phần của đĩa vật lý C được đặt với các tên là D,E,F,G,H... Việc phân chia một đĩa vật lý thành nhiều đĩa Logic được thực hiện khi gọi lệnh ngoại trú FDISK của DOS.

2. Quy định tên thư mục và tên tệp

Các văn bản hoặc chương trình do người sử dụng lập, được lưu trữ trong máy dưới các tên khác nhau để khi cần có thể gọi ra xử lý. Mỗi văn bản hoặc chương trình được lưu trên đĩa dưới dạng một Tệp (FILE). Tên tệp theo quy định của DOS gồm 2 phần:

- Phần tên được chọn nhiều nhất 8 ký tự,

- phần đuôi gồm không quá 3 ký tự,

Các ký tự có thể là chữ cái hoặc chữ số và phải viết liền nhau. Giữa tên và đuôi phân cách bằng một dấu chấm, ví dụ: COMMAND.COM, THO.VNS, AUTOEXEC.BAT, CONFIG.SYS v.v... Khi đặt tên tệp máy không phân biệt chữ thường hay chữ hoa, nếu tên tệp dài quá 8 ký tự máy sẽ tự động bỏ các ký tự từ ký tự thứ 9 trở đi.

Thư mục (DIRECTORY) được sử dụng để quy ước một phần không gian trên đĩa cho việc lưu trữ một số văn bản hoặc chương trình mà người sử dụng lập ra. Nếu nhiều người cùng sử dụng chung một ổ đĩa thì thư mục có tác dụng bảo vệ tránh sự truy nhập hoặc xử lý nhầm dữ liệu của người khác.

Tên thư mục được đặt nhiều nhất là 11 ký tự và không có phần đuôi như tên tệp. Các ký tự sử dụng đặt tên cho thư mục cũng giống như các ký tự dùng đặt tên cho tệp.

Chú ý:

Khi đặt tên tệp hoặc thư mục không được sử dụng các ký tự sau đây: . , . ? \ /;: + = < > [ ]

Trong MS-DOS thư mục được tổ chức dưới dạng cây. Thư mục gốc (Root directory) được quy định là đĩa hiện thời, thư mục gốc được tạo ra ngay sau khi format đĩa. Trong các lệnh của DOS thư mục gốc được ký hiệu bởi dấu gạch chéo ngược " \ ".

Dưới Root Directory là các thư mục con (Subdirectory), đối với đĩa mật độ thường (360 Kb) thì mỗi Rootdir có thể chứa được 112 Subdir, còn với các đĩa mật độ cao số lượng Subdir có thể tạo ra là 224. Trong mỗi Subdirectory ta lại có thể tạo ra các thư mục ở mức thấp hơn giống như sự phân nhánh trên cành cây. Cần lưu ý rằng khi đang làm việc trong một Subdir nào đó thì DOS không quan tâm đến các Subdir khác, nghĩa là các tệp thuộc thư mục khác dù trùng tên với tệp thuộc thư mục hiện thời cũng không bị gọi ra xử lý nếu câu lệnh DOS không chỉ rõ đường dẫn cụ thể.

Để tạo lập hoặc tìm kiếm tệp trên một ổ đĩa có cấu trúc thư mục hình cây, DOS cần phải biết đường dẫn (PATH), đường dẫn được hợp thành bởi ba yếu tố sau đây:

- **Tên ổ đĩa**

- **Tên thư mục** (và các thư mục bậc thấp hơn, nếu có)

- **Tên tệp** (kể cả đuôi)

Đường dẫn là một phần không thể thiếu trong các câu lệnh của DOS, nó là một dãy các ký tự đặt sau từ khoá lệnh (nội trú hoặc ngoại trú) biểu thị tên ổ đĩa, tên các thư mục con, giữa các thư mục con được ngăn cách bởi dấu "\", cuối của đường dẫn là tên tệp.

3. Mô tả tệp

Trước khi làm quen với việc mô tả một tệp, ta cần biết đến một quy định khác cúa DOS đó là dấu nhắc hệ thống. Sau khi khởi động máy, với những máy khởi động bằng đĩa mềm thì dấu nhắc hệ thống là " A> ", còn máy khởi động từ ổ cứng thì dấu nhắc sẽ là " C> ". Dấu nhắc này thường gây khó khăn cho người sử dụng khi gõ lệnh chuyển vào thư mục do vậy người ta thường dùng dấu nhắc hệ thống dạng "A:\>" hoặc C:\>. Để tạo ra dấu nhắc kiểu này trong tệp AUTOEXEXC. BAT ta cần phải đưa vào lệnh PROMPT PG. Trong phần sau của tài liệu sẽ đề cập cụ thể hơn việc tạo lập tệp AUTOEXEC.BAT và ý nghĩa của nó.

Để mô tả một tệp cần phải chỉ rõ tên ổ đĩa, tên thư mục chứa tệp và tên tệp kể cả phần đuôi.

Ví dụ: để chép tệp CONGVAN.VNS trong thư mục HANHCHINH ở ổ đĩa A sang ổ đĩa B ta viết câu lệnh như sau:

A:\ COPY A:\HANHCHINH\CONGVAN.VNS B:↵

III. CÁC LỆNH CỦA HỆ ĐIỀU HÀNH MS-DOS

1. Nhóm lệnh về thư mục

1.1 Lệnh xem thư mục DIR

Lệnh DIR (viết tắt của từ Directory) dùng để xem trong thư mục gốc hoặc trong thư mục con có những tệp nào đang được lưu trữ. Giả sử vị trí hiện thời là ở thư mục gốc trên ổ C, để xem đĩa trên ổ A có những thư mục hoặc tệp gì ta gõ:

C:\>DIR A: ↵

Chú ý:

Ký hiệu ↵ trong các lệnh của DOS là quy ước yêu cầu người sử dụng phải bấm phím Enter.

Để xem nội dung lưu trữ trong một thư mục sau lệnh DIR cần phải chỉ rõ ổ đĩa chứa thư mục, tên thư mục và loại tệp cần tìm (thể hiện qua phần đuôi). Ngoài ra còn có thể dùng thêm các tham số /w hoặc /p để hiển thị trên màn hình theo chiều ngang hay theo chiều dọc. Ví dụ từ ổ đĩa chủ C cần xem trong thư mục BKED ở ổ đĩa D có những tệp nào có đuôi là VNS, lệnh viết như sau:

C:\>DIR D:\BKED*.VNS ↵

Tên tệp sau lệnh DIR có thể viết đầy đủ hoặc sử dụng các ký hiệu " * " và " ? ". Ký tự "* " dùng để thay cho một nhóm ký tự đầu hoặc đuôi tệp, ký tự " ? " dùng để thay cho một ký tự bất kỳ nào mà ta không muốn chỉ định cụ thể trong tên tệp.

Ví dụ : lệnh xem nội dung thư mục BKED trong ổ đĩa D:

C:\>DIR D:\BKED*.* ↵

Lệnh tìm các tệp có hai ký tự đầu là QN trong thư mục BKED

C:\>DIR D:\BKED\QN??????.* ↵ hoặc

C:\>DIR D:\BKED\QN*.* ↵

Trong lệnh trên ta phải viết đủ sáu dấu "?" vì tên tệp trong trường hợp tổng quát có 8 ký tự, giả sử ta đã biết chắc tên các tệp chỉ có 4 ký tự thì sau hai ký tự QN chỉ cần viết 2 dấu "?". Câu lệnh trên cũng có thể viết dưới dạng thứ 2 như sau C:\>DIR D:\BKED\QN*.* ↵

** Lệnh DIR mở rộng*

Sau lệnh DIR nội dung của thư mục hoặc ổ đĩa sẽ hiện lên màn hình, vì màn hình chỉ bố trí 25 dòng nên tên các tệp hoặc thư mục sẽ trôi từ dưới lên với tốc độ khá nhanh khiến ta không thể đọc kịp. Muốn đọc ta có thể chỉ định một trong hai tham số /W hoặc /P. Tham số /P cho hiện lên từng trang màn hình theo chiều dọc, bấm phím bất kỳ sẽ hiện lên trang tiếp theo, với lệnh này ta có thể đọc được ngày giờ tạo ra tệp và dung lượng tính theo byte. Tham số /W cho hiện lên tất cả theo chiều ngang và không có ngày giờ cùng dung lượng tệp.

Ví dụ: C:\>DIR A:\BKED*.* /W ↵

1..2 Lệnh tạo thư mục MD (make Directory)

Trên một đĩa có thể lưu trữ nhiều loại dữ liệu khác nhau, để dễ tìm kiếm mỗi loại dữ liệu nên để trong một thư mục với tên đặc trưng, ví dụ các thư mục CONGVAN, THUTU, QU_DINH,... Lệnh tạo thư mục được viết như sau

MD tên thư mục (tên thư mục viết liền và không có dấu). Để tạo thư mục CONGVAN trên đĩa mềm ở ổ A ta gõ lệnh:

A:\>MD CONGVAN ↵

Để tạo thư mục ở mức thấp hơn, ví dụ thư mục DOINGOAI trong thư mục CONGVAN ta có thể chuyển vào thư mục CONGVAN rồi gõ lệnh: MD DOINGOAI hoặc gõ trực tiếp ngay sau dấu nhắc hệ thống:

A:\>MD A:\CONGVAN\DOINGOAI ↵

1.3 Lệnh chuyển vào thư mục CD (change Directory)

Để thâm nhập vào trong một thư mục ta dùng lệnh CD, cú pháp của lệnh này như sau:

CD tên thư mục ↵

Để vào thư mục CONGVAN ta viết lệnh CD CONGVAN ↵

1.4 Lệnh ra khỏi thư mục

Để ra khỏi thư mục chia thành hai trường hợp:

* Ra khỏi thư mục hiện thời trở về thư mục gốc: CD\

A:\CONGVAN\DOINGOAI>CD\ ↵

* Ra khỏi thư mục con trở về thư mục ở mức cao hơn

A:\CONGVAN\DOINGOAI>CD.. ↵

Sau lệnh này dấu nhắc hệ thống sẽ là A:\CONGVAN>

Chú ý:

Không thể dùng lệnh CD tên thư mục để nhảy từ thư mục bậc thấp lên thư mục bậc cao hơn nó mà cần phải về thư mục gốc sau đó mới chuyển vào thư mục cần tìm.

1.5 Lệnh xoá thư mục RD (remove Directory)

Để xoá một thư mục cần phải bảo đảm đầy đủ hai yếu tố sau

* Thư mục đó là rỗng nghĩa là không còn bất kỳ một tệp nào trong thư mục định xoá.

* Vị trí con trỏ hiện thời phải nằm ở mức cao hơn so với thư mục cần xoá, ví dụ cần xoá thư mục DOINGOAI trong thư mục con CONGVAN ta gõ lệnh:

A:\>RD \CONGVAN\DOINGOAI ↵.

Trong trường hợp các tệp đã được gán thuộc tính chỉ đọc (read only) thì phải dùng các lệnh của DOS hoặc của một phần mềm ứng dụng khác như PCTOOLL, NORTON... để xoá thuộc tính đó đi trước khi có thể làm rỗng thư mục bằng các lệnh xoá tệp.

2. Nhóm lệnh về tệp

2.1 Lệnh hiển thị nội dung tệp

Nội dung của các tệp văn bản cất giữ trong đĩa có thể gọi hiển thị trên màn hình bởi lệnh TYPE, cú pháp của lệnh này như sau:

TYPE tên tệp (cả đuôi) ↵

Những tệp có đuôi COM, EXE, FOX... là những tệp chương trình đã được biên dịch do đó sau lệnh TYPE trên màn hình sẽ hiện lên các ký hiệu mà ta không thể đọc được.

Chú ý:

Những tệp có độ dài lớn hơn 25 dòng khi hiện trên màn hình các dòng sẽ trôi từ dưới lên trên, muốn các dòng dừng lại để đọc ta bấm phím PAUSE, đọc xong bấm phím bất kỳ để hiện tiếp các dòng còn lại.

2.2 Lệnh đổi tên tệp REN (rename)

Để đổi tên tệp cần phải chỉ rõ đường dẫn từ vị trí hiện thời tới nơi lưu trữ tệp. Cú pháp của lệnh như sau:

REN <đường dẫn> tên tệp cũ, tên tệp mới.

Ví dụ ta đang ở ổ đĩa C: cần đổi tên tệp THU.VNS trong thư mục CONGVAN ở ổ A: sang tên mới là NHAP.VNS ta gõ lệnh:

C:\>REN A:\CONGVAN\THU.VNS NHAP.VNS ↵

Chú ý: *Lệnh đổi tên tệp không dùng để đổi tên thư mục*

2.3 Lệnh xoá tệp DEL (Delete)

Để xoá tệp ta chỉ cần thực hiện lệnh xoá tên tệp. Cần hết sức thận trọng khi thực hiện lệnh xoá tệp, tránh xoá nhầm các tệp mà ta đang sử dụng. Tốt nhất không nên viết trong lệnh đường dẫn tới tệp mà dùng lệnh chuyển thư mục CD để vào hẳn trong thư mục, sau khi đã ở trong thư mục rồi ta viết lệnh với cú pháp sau:

DEL tên tệp ↵

Ví dụ: A:\CONGVAN>DEL THU.VNS ↵

Muốn xoá tất cả các tệp có trong thư mục ta gõ lệnh

A:\CONGVAN> DEL *.* ↵

Khi đó trên màn hình xuất hiện thông báo:

All files in directory will be deleted. Are you sure (Y/N)?

Các tệp trong thư mục sẽ bị xoá. Bạn có chắc chắn muốn xoá hay không? Nếu gõ Y tệp sẽ bị xoá, nếu gõ N lệnh sẽ không được thực hiện.

2.4 Lệnh xóa tệp ERASE

Lệnh ERASE cũng có tác dụng xoá tệp như lệnh DEL, cú pháp của lệnh này có thể xem thêm trong lệnh DEL.

2.5 Lệnh chép tệp COPY

Để chép tệp từ đĩa này sang đĩa khác hoặc từ thư mục này sang thư mục khác ta dùng lệnh COPY, tuỳ theo vị trí của tệp nguồn và tệp đích hoặc đĩa nguồn và đĩa đích mà câu lệnh sẽ được viết cụ thể:

Ví dụ 1:

A:\CONGVAN> COPY B:\HOPDONG\THU.BK ↵

Trong ví dụ trên ta đang ở thư mục CONGVAN ở ổ đĩa A lệnh copy chép cho ta tệp THU.VNS trong thư mục HOPDONG ở ổ B vào thư mục CONGVAN ở ổ A.

Ví dụ 2:

C:\>COPY C:\BKED\NGOC4.MNU A:\CONGVAN\KIEMTRA.AAA ↵

Trong ví dụ 2 ta đã thực hiện việc chép tệp NGOC4.MNU trong thư mục BKED trên ổ đĩa C sang thư mục CONGVAN trên ổ đĩa A đồng thời đổi tên tệp thành KIEMTRA.AAA

Chú ý:

1. Nếu muốn chép tệp nguồn sang tệp đích mà lại cùng ở trong một thư mục thì tệp đích phải lấy tên khác tên tệp nguồn. Nếu tệp nguồn và tệp đích ở trên cùng một đĩa nhưng trong hai thư mục khác nhau thì có thể giữ nguyên tên tệp khi thực hiện lệnh COPY.

2. Nếu tệp nguồn nằm trên thư mục gốc thì thì phải dùng lệnh COPY theo mẫu trong ví dụ 2

2.6 Ghép nối các tệp

Sử dụng lệnh COPY ta có thể ghép nối các tệp với nhau. Giả sử ta có hai tệp A1.VNS và A2.VNS ta cần ghép chúng thành tệp A3.VNS, lệnh sẽ được viết như sau:

C:\>COPY A1.VNS+A2.VNS A3.VNS ↵

Chú ý:

1. Nếu không đưa ra tên tệp đích (A3.VNS) thì lệnh COPY sẽ nối tệp A2.VNS vào tệp A1.VNS và nội dung của A1.VNS bây giờ sẽ bao gồm cả A2.VNS

2. Nếu các tệp A1.VNS và A2.VNS nằm trong các ổ đĩa khác nhau hoặc các thư mục khác nhau thì phải chỉ rõ đường dẫn trong lệnh COPY như đã trình bày trong phần đầu.

3. Lệnh in ra máy in

Những lệnh in sau đây khi thực hiện chỉ in được các ký tự tiếng Anh, những ký tự tiếng Việt sẽ không xuất hiện được đúng với dạng chuẩn của nó. Có hai dạng lệnh in có thể sử dụng:

Sử dụng lệnh TYPE - Giả sử cần in nội dung tệp BAOCAO.WP trong thư mục CONGVAN trên ổ đĩa A ra máy in ta gõ lệnh:

A>TYPE A: \CONGVAN\BAOCAO.WP>PRN ↵

Sử dụng lệnh COPY

A>COPY A:\CONGVAN\BAOCAO.WP PRN ↵

Với các văn bản tiếng Việt soạn thảo bằng BKED, VNI, VIETSTAR,... muốn in ra ta phải sử dụng các lệnh in của chính các chương trình đó . Những lệnh in này không thể thực hiện từ mức hệ điều hành DOS.

4. Lệnh về thời gian

Trong máy tính có một lịch tự động chỉ thời gian và chúng ta có thể bất kỳ lúc nào hỏi máy về giờ giấc và ngày tháng.

* Để xem giờ từ dấu nhắc hệ thống ta gõ lệnh:

C:\>TIME ↵

Màn hình xuất hiện dòng chữ: Current time is: 3:28:15.40P Enter new time:

Nếu muốn thay đổi giờ thì ta điền vào sau dòng chữ Enter new time giờ mới, nếu khômg muốn thay đổi thì bấm phím Enter

* Để xem ngày tháng ta gõ

C:\>DATE ↵

Màn hình xuất hiện dòng chữ:

Current date is Wes 10-28-1992 Enter new date (mm-dd-yy):

Dòng trên cho biết ngày hiện thời theo thứ tự là thứ mấy, tháng, ngày, năm. Nếu muốn thay đổi ta chỉ điền vào hai chữ số theo mẫu ở dòng dưới theo thứ tự Tháng-Ngày-Năm.

5. Lệnh khởi tạo đĩa - FORMAT

Lệnh này dùng để khởi tạo một đĩa trắng (hoặc khởi tạo lại một đĩa đang dùng nhưng bị hỏng) trong một ổ đĩa được chỉ định trước khi đưa đĩa vào sử dụng. Muốn thực hiện lệnh FORMAT, trong đĩa hệ thống (system disk) phải có tệp FORMAT. COM. Giả sử hệ điều hành được cài đặt ở ổ C và ta cần khởi tạo một đĩa trên ổ A. Lệnh FORMAT có thể có một hoặc nhiều tham số ở đây chỉ nêu ra ba tham số hay dùng:

C:\>FORMAT A:/v /u /s ↵

Tham số /v báo cho DOS biết ta cần đặt tên cho đĩa đang được FORMAT

Tham số /u báo cho DOS format vô điều kiện, không cần cứu các dữ liệu sau khi đã format

Tham số /s báo cho DOS biết đĩa đang FORMAT sẽ được dùng làm đĩa hệ thống và DOS có trách nhiệm chép các tệp hệ thống từ ổ đĩa chủ sang đĩa đang format theo thứ tự: IO.SYS, MSDOD.SYS, COMMAND.COM.

Chú ý:

** Những đĩa đang dùng nếu tiến hành FORMAT thì toàn bộ dữ liệu lưu trên đĩa sẽ bị mất, do đó cần chép bảo toàn dữ liệu trước khi tiến hành FORMAT.*

** Không lắp các đĩa có dung lượng khác với dung lượng của ổ khi thực hiện lệnh FORMAT.*

** Để khởi tạo đĩa cứng có thể dùng lệnh FDISK của DOS tuy nhiên nếu người sử dụng chưa nắm chắc về DOS thì không nên tự mình khởi tạo đĩa cứng để tránh những sự cố đáng tiếc có thể xảy ra.*

6. Lệnh đặt tên đĩa

Nếu ta Format đĩa với tham số V thì có thể đặt tên cho đĩa trong quá trình sử dụng muốn đổi tên đĩa ta gõ lệnh LABEL, cú pháp của lệnh này như sau:

C:\>LABEL [D:]Tên đĩa ↵

Trong lệnh trên [D:] là tên ổ chứa đĩa cần đổi tên (chứ không phải bắt buộc là ổ D), còn " Tên đĩa " là tên mà ta muốn đặt cho đĩa. Tên đĩa có thể đặt nhiều nhất là 11 ký tự và có thể có dấu cách ở giữa. Nếu sau lệnh LABEL ta chưa viết Tên đĩa thì máy sẽ nhắc

VOLUME IN DRIVE D: IS XXXXXXXXX (tên đĩa cũ trên ổ là ....

VOLUME LABEL (11 CHARACTERS, ENTER FOR NONE)? (đặt tên mới cho đĩa,11 ký tự, không đặt tên thì nhấn Enter)

Những đĩa khi Format chưa đặt tên thì LABEL sẽ đưa ra dòng thông báo:

VOLUME IN DRIVE D: HAS NO LABEL (đĩa chưa đặt tên)

VOLUME LABEL (11 CHARACTERS, ENTER FOR NONE)? (đặt tên mới cho đĩa, 11 ký tự, không đặt thì nhấn Enter)

7. Lệnh hiển thị tên đĩa VOL

Để đọc tên đĩa ta dùng lệnh VOL với cú pháp sau:

C:\>VOL [D:] ↵

Nếu đĩa chưa được đặt tên thì máy sẽ có thông báo:

VOLUME IN DRIVE ... HAS NO LABEL.

8. Lệnh chép đĩa DISKCOPY

DISKCOPY là một lệnh rất thuận tiện cho phép sao chép toàn bộ nội dung một đĩa sang đĩa khác kể cả trường hợp đĩa được sao là đĩa hệ thống. Các bước của thao tác như sau:

Đặt đĩa nguồn (đĩa cần sao) vào một ổ có dung lượng phù hợp, ví dụ ổ A. Đặt đĩa đích vào ổ B sau đó gõ lệnh:

C:\>DISKCOPY A: B : ↵

Nếu đĩa đích đã được Format trước khi đặt vào ổ thì toàn bộ nội dung của đĩa trên ổ A sẽ được chép sang ổ B. Nếu đĩa trên ổ A là một đĩa hệ thống thì ta có một bản lưu đĩa hệ thống để đề phòng trường hợp đĩa hệ thống đang dùng có sự cố.

Trong trường hợp đĩa đích là đĩa mới chưa được Format thì DISKCOPY sẽ tự động Format đĩa này theo các tiêu chuẩn của đĩa nguồn trước khi sao chép.

Chú ý:

1- Giữa đĩa cứng và đĩa mềm không dùng được lệnh DISKCOPY

2- Nếu đĩa nguồn và đĩa đích được tạo khuôn theo các tiêu chuẩn khác nhau thì lệnh DISKCOPY không thực hiện được.

3- Nếu một trong hai đĩa có một rãnh bị hỏng thì DISKCOPY không làm việc.

4- Nếu máy chỉ có một ổ đĩa mềm thì vẫn có thể tiến hành việc sao chép đĩa với cú pháp lệnh

C:\>DISKCOPY A: A: ↵

Trong trường hợp này máy sẽ nhắc nhở ta lắp đĩa nguồn và đĩa đích vào ổ A tại từng thời điểm thích hợp. Sau khi sao chép xong, màn hình xuất hiện thông báo:

COPY COMPLETE

COPY ANOTHER (Y/N)?

Nếu muốn Copy tiếp ta gõ Y, nếu kết thúc lệnh DISKCOPY ta gõ N.

9. Lệnh tạo bản sao tệp hoặc thư mục

Cú pháp: BACKUP <vị trí nguồn> <vị trí đích> [<các tham số>]

Lệnh Backup sao lưu dự phòng các tệp từ đĩa này sang đĩa khác không hạn chế đĩa đó là cứng hay mềm.

- Vị trí nguồn: Bao gồm đường dẫn, tên ổ, tên thư mục, tên tệp. Nếu chỉ có tên ổ thì toàn bộ nội dung ổ đĩa sẽ được sao lưu, trong trường hợp sao lưu ổ cứng có dung lượng lớn thì DOS sẽ yêu cầu lắp lần lượt từng đĩa mềm vào ổ mềm và sao lại cho đến khi kết thúc.

- Vị trí đích: Chỉ định vị trí ổ mà ta muốn ghi các tệp dự phòng lên đó.

- Các tham số: /S Lưu dự phòng cả nội dung của cả các thư mục con

/a Lưu dự phòng lên một đĩa đã có các tệp tin dự phòng được lưu trước mà không xoá các tệp đó

10. Chép các tệp dự phòng sang nơi khác

Trên đĩa dự phòng ta không đọc được từng tên tệp cụ thể , tất cả các tệp dự phòng đều được ghi chung vào một tệp trên đĩa dự phòng với tên lần lượt là BACKUP01, BACKUP02,...

Để khôi phục lại các tệp đó ta dùng lệnh:

RESTORE <ổ đĩa 1> <ổ đĩa 2> <thư mục> [<tham số>]

- ổ đĩa 1: Nơi lưu trữ các tập tin dự phòng

- ổ đĩa 2: Chỉ định ổ đĩa nơi các tệp tin dự phòng sẽ được phục hồi lên.

- Thư mục: tên thư mục mà ta dự định chép các tệp tin dự phòng vào.

Tên thư mục này cần trùng với tên thư mục khi ta sử dụng lệnh BACKUP

- Tham số: /s phục hồi mọi thư mục con

/p phục hồi cả các tệp có thuộc tính chỉ đọc

11. Lệnh thiết lập chế độ in đồ hoạ

Để có thể in các hình ảnh trên màn hình ra máy in bằng phím Print Screen ta có thể gọi một lệnh ngoại trú của DOS. Cú pháp:

C:\DOS>GRAPHICS ↵

Sau lệnh này nếu trên màn hình có một ảnh thì ảnh đó có thể in ra giấy bằng cách bấm phím Print Screen

12. Lệnh nhắc lại các lệnh đã gõ

Để lưu lại các lệnh đã gõ và có thể gọi ra bằng các phím dịch chuyển con trỏ cần phải đưa vào thường trú lệnh ngoại trú DOSKEY.COM. Lệnh này được viết thành một dòng trong tệp AUTOEXEC.BAT như sau:

C:\DOS\DOSKEY.COM

13. Lệnh gán thuộc tính

Các tệp (văn bản hoặc chương trình) lưu trữ trên máy có 4 thuộc tính là :

- Read only : thuộc tính chỉ đọc (không thể ghi thêm gì vào tệp có thuộc tính này)

- Archive : thuộc tính lưu trữ (tất cả mọi tệp đều có)

- Hiden : thuộc tính ẩn (các tệp bị giấu không thấy tên trong lệnh DIR)

- System : thuộc tính hệ thống (chỉ có đối với một số tệp cụ thể)

Để gán (hoặc bỏ) thuộc tính cho một tệp nào đó ta dùng lệnh ATTRIB kèm theo tham số là chữ cái đầu cúa các thuộc tính r, a, h, s. Trước các chữ cái này ta đặt dấu + nếu là gán thuộc tính và dấu - nếu bỏ thuộc tính đã gán.

Ví dụ : Gán cho tệp BAITAP.VNS thuộc tính chỉ đọc và ẩn

A:\>ATTRIB BAITAP.VNS +h ↵

A:\>ATTRIB BAITAP.VNS +r ↵

14. Lệnh hiện cây thư mục TREE

Cú pháp : TREE [drive] [path]

. Drive : chọn ổ đĩa để xem cây thư mục

. Path : chọn thư mục để xem

- Lệnh TREE không tham số sẽ cho hiện cây thư mục chỉ gồm tên các thư mục trên ổ đĩa hiện thời, lệnh này khác lệnh DIR ở chỗ nó không cho ta biết gì thêm về dung lượng, ngày tạo thư mục ...

- Để xem thư mục kèm theo các tệp hiện có trong thư mục và cho hiện mỗi lần một màn hình ta dùng lệnh TREE kèm theo tham số như sau:

C:\>TREE C:\ /f |more

15. Lệnh trợ giúp HELP

Trong trường hợp ta chưa hiểu rõ lắm về một lệnh nào đó của DOS có thể dùng lệnh HELP kèm theo tên lệnh cần tìm hiểu DOS sẽ cho hiện lên màn hình những chỉ dẫn liên quan đến lệnh ta quan tâm. Các chỉ dẫn này bao gồm cú pháp lệnh, các tham số kèm theo lệnh và các lệnh khác của DOS có liên quan đến lệnh đang tìm hiểu

Ví dụ : cần tìm hiểu về lệnh COPY:

A:\> HELP COPY ↵

IV. TỆP AUTOEXEC.BAT VÀ TỆP CẤU HÌNH CONFIG.SYS

1. Tệp cấu hình CONFIG.SYS

Tệp cấu hình CONFIG.SYS là một tệp đặc biệt do người sử dụng thiết lập để thay đổi các thông số cấu hìmh hệ thống. Mỗi khi khởi động DOS sẽ tự động tìm tệp CONFIG.SYS trên đĩa hệ thống thực hiện các lệnh trong tệp này. Nếu CONFIG.SYS chưa được lập thì DOS sẽ gán các giá trị ngầm định (default values) cho các lệnh về cấu hình. Giả sử ổ đĩa chủ là ổ C, để tạo tệp CONFIG. SYS ta gõ từ bàn phím lệnh sau:

C:\>copy con config.sys(E)

Tiếp tục gõ các lệnh về cấu hình mà ta muốn có trong tệp, sau mỗi dòng lệnh cần ấn Enter. Khi vào xong các lệnh ấn phím F6 rồi ấn Enter để kết thúc công việc và ghi tệp lên đĩa.

Có nhiều lệnh có thể đưa vào trong tệp CONFIG.SYS ở đây ta chỉ nghiên cứu một số lệnh hay dùng:

1.1 Lệnh COUNTRY

Lệnh này dùng để biểu diễn ngày, tháng, giờ, tiền tệ theo quy định của một nước nào đó. Cú pháp của lệnh như sau:

COUNTRY = xxx,

xxx là mã số của quốc gia dùng trong mạng bưu điện quốc tế, ví dụ:

Mỹ 001 Anh 044 Pháp 033 Italia 039

v.v ....

Nếu ta không ghi lệnh COUNTRY thì DOS sẽ tự động gán cho nó giá trị ngầm định là 001, nghĩa là ngày tháng được viết dưới dạng: mm-dd-yy (tháng-ngày-năm), giờ thì có dạng hh:mm:ss (giờ: phút:giây), đơn vị tiền tệ là $ (dollar), dấu ngăn thập phân là dấu chấm . Nước ta ngày tháng không viết kiểu Mỹ do vậy lệnh COUNTRY nên chọn tham số là 044.

1.2 Lệnh FILES

Lệnh này cho phép ta quy định số tệp lớn nhất có thể mở đồng thời để làm việc, cú pháp lệnh như sau:

FILES = xxx

trong đó xxx lấy giá trị từ 8 đến 255, giá trị ngầm định là 8. Nếu máy làm việc với FOXBASE hoặc FOXPRO thì FILES nên lấy bằng 15-20.

1.3 Lệnh BUFFERS

Lệnh này quy định số vùng đệm dành cho đĩa trong bộ nhớ sau khi khởi động máy, lệnh viết như sau: BUFFERS = xx,

Ở đây xx lấy giá trị từ 1 đến 99, giá trị ngầm định là 3. Vùng đệm là vùng mà DOS sử dụng để lưu nội dung của một sector nào đó khi chương trình có lệnh đọc nó từ đĩa vào. Sau đó chương trình sẽ chuyển phần nội dung cần thiết vào vùng nhớ mà nó quản lý. Nếu sau đó lại có lệnh đọc hoặc ghi thì trước hết DOS sẽ tìm trong vùng đệm, nếu nội dung cần thiết đã có thì nó chuyển thẳng vào vùng nhớ mà không đọc trên đĩa nữa, điều này cho phép tiết kiệm thời gian một cách đáng kể. Với những máy có cài đặt FOXBASE, DB3, FOXPRO, ... thì vùng đệm cần chọn khoảng từ 10 đến 20. Không nên chọn quá nhiều vùng đệm vì mỗi vùng đệm phải chứa được một sector nghĩa là cần đến 512 bytes trong bộ nhớ.

1.4 Lệnh DEVICE

Lệnh này cho phép cài đặt thêm các chương trình điều khiển ngoại vi ngoài các chương trình điều khiển chuẩn như bàn phím, màn hình, máy in, đĩa mềm, đĩa cứng ... mà DOS đã nạp ngay khi khởi động máy . Hai chương trình điều khiển ngoại vi mà DOS cung cấp cho người sử dụng là: ANSI.SYS và RAMDRIVE.SYS.

ANSI.SYS là chương trình mở rộng chế độ của màn hình, còn RAMDRIVE. SYS là chương trình tạo lập đĩa ảo. Đĩa ảo thực chất là một vùng của bộ nhớ được mô phỏng như một đĩa từ. Nếu đĩa ảo đã được tạo thì ta có thể sao thông tin vào đó và quá trình xử lý sẽ diễn ra nhanh hơn nhiều lần so với xử lý thông tin trên đĩa mềm. Muốn cài đặt một đĩa ảo trong tệp CONFIG.SYS ta phải đưa vào lệnh sau:

DEVICE = [D:][path]RAMDRIVE.SYS [bbb][sss][ddd][/E]

[D:] và [path] là tên ổ đĩa và thư mục chứa tệp RAMDRIVE.SYS, bbb là kích thước của đĩa ảo tính bằng Kbytes, ngầm định là 64, sss là kích thước của sector, có thể chọn bằng 128, 256, hoặc 512 bytes, ngầm định là 128, ddd là số danh mục chứa trong Directory, có thể lấy từ 2 đến 512, ngầm định là 64. /E là tham số quy định đặt đĩa ảo trong phần mở rộng của bộ nhớ, điều này thường tiến hành khi bộ nhớ có dung lượng lớn hơn 1 MB.

Ví dụ:

DEVICE=[A:][DOS33]RAMDRIVE.SYS 360 512 100

Trong ví dụ trên tệp RAMDRIVE.SYS nằm trong thư mục DOS33 ở ổ đĩa A, ta đã tạo ổ đĩa ảo có dung lượng 360 K, mỗi sector là 512 bytes, và số danh mục là 100.

2. Tệp AUTOEXEC.BAT

Khi làm việc trên máy tính ta có thể gặp một số lệnh lặp đi lặp lại, để tiết kiệm thời gian ta có thể gộp các tệp đó vào một tệp đặc biệt gọi là tệp lô (BATCH). Khi cần gọi các lệnh đó ta chỉ việc gọi tên tệp là đủ. Tất cả những tệp thuộc dạng BATCH đều có đuôi là BAT, tệp AUTOEXEC.BAT cũng là một tệp thuộc dạng này. Tệp AUTOEXEC.BAT luôn được đặt trong thư mục gốc của đĩa hệ thống, khi khởi động sau khi đọc các thông số từ ROM BIOS DOS bao giờ cũng tìm tệp này và thực hiện các lệnh có trong đó, lúc này trên màn hình sẽ không có thông báo về ngày giờ nữa. Việc tạo ra tệp AUTOEXEC. BAT có thể được thực hiện bằng lệnh COPY CON như đối với tệp CONFIG.SYS hoặc bằng bất kỳ một phần mềm soạn thảo nào khác ví dụ EDLIN, BKED ,...

Với những máy khởi động trên đĩa mềm tệp AUTOEXEC.BAT được cài vào thư mục gốc trên đĩa còn với máy khởi động trên đĩa cứng thì tệp này được cài vào ổ đĩa C. Thông thường trong tệp AUTOEXEC.BAT người ta hay đưa vào đó lệnh định nghĩa dấu nhắc của DOS tức là lệnh PROMPT PG sau đó là các lệnh truy nhập vào một chương trình nào đó cần làm ngay khi khởi động máy ví dụ lệnh dò tìm Virus hoặc lệnh diệt Virus.

Ngoài ra trong tệp AUTOEXEC. BAT người ta cũng hay đưa vào lệnh về đường dẫn PATH để có thể truy nhập đến các tệp thi hành mà không cần lệnh chuyển thư mục. Ngoài tệp AUTOEXEC. BAT những người làm việc thường xuyên trên máy tính cũng hay tạo cho mình những tệp BAT khác để giảm bớt thời gian gõ lệnh trên bàn phím. Ví dụ để khởi động chương trình BKED từ ổ đĩa A thông thường ta phải gõ các lệnh:

A:\>CD BKED ↵

A:\BKED>BKED ↵

Ta có thể tạo tệp BK.BAT bằng các lệnh sau:

A:\>COPY CON BK.BAT ↵

CD BKED ↵

BKED.EXE ↵

Nhấn phím F6 rồi ↵

Từ đây về sau để khởi động chương trình BKED từ dấu nhắc của DOS ta chỉ việc gõ BK (E) .

3. Một số lưu ý đặc biệt đối với người sử dụng

Ở mức hệ điều hành DOS chúng ta chỉ quản lý được các ổ đĩa, các thư mục và các tệp. Muốn tạo lập các tệp mới, thay đổi nội dung các tệp cũ cần phải sử

dụng các chương trình soạn thảo văn bản. Chương trình soạn thảo văn bản trong DOS có tên là EDLIN, các phần mềm khác như TURBO PASCAL, FOXPRO PRO,... cũng đều có phần soạn thảo để người sử dụng tạo lập, sửa chữa các văn bản hoặc chương trình nguồn, ta sẽ không đề cập đến những điều đó trong phần này.

Nhằm tạo điều kiện thuận tiện cho người sử dụng dấu nhắc hệ thống trong DOS thường được viết: " C:\> ", với kiểu dấu nhắc này sau khi dùng lệnh vào thư mục CD thì tên thư mục sẽ được hiện trên màn hình ở vị trí của dấu nhắc, ví dụ:

C:\congvan>_

Những lệnh trình bày trên đây mới chỉ là một phần rất nhỏ trong tổng số lệnh của DOS. Ngay với hệ điều hành MS-DOS Version 5.0 ta cũng đã thấy rằng hệ lệnh của DOS là hết sức phong phú và cả những người làm máy tính chuyên nghiệp cũng không thể nhớ hết được nội dung của cuốn sách dày 796 trang này . Tuy vậy chỉ với những lệnh đã được trình bày nếu chúng ta sử dụng thành thạo hết mọi tính năng của chúng thì cũng đáp ứng được các yêu cầu quản lý và sử dụng máy vi tính .

Ghi chú :

Các lệnh trình bày trên đây được trích ra từ bộ lệnh của MS-DOS Version 5.0 tuy nhiên các Version của DOS bao giờ cũng có tính kế thừa do vậy nếu máy tính của bạn có DOS 6.0 hoặc 7.0 thì bạn cũng vẫn có thể yên tâm gõ các lệnh đã biết mà không cần thêm bất cứ tham số gì.

B. HỆ ĐIỀU HÀNH WINDOWS

I. KHÁI QUÁT VỀ HỆ ĐIỀU HÀNH WINDOWS

Hệ điều hành Windows là hệ điều hành sử dụng công cụ đồ họa, thay vì phải viết các lệnh dài dòng người sử dụng chỉ cần bấm chuột vào các biểu tượng. Các phiên bản đầu tiên của Windows là Windows 3.1, Windows95 chưa phải là hệ điều hành hoàn thiện, chỉ đến Windows 2000 nó mới trở thành một hệ điều hành thực thụ. Các ưu điểm nổi bật của Windows là:

- Cung cấp một giao diện đồ hoạ GUI - Graphic User Interface

- Tự động nhận dạng và cài đặt trình điều khiển các thiết bị (Plus and Play).

- Hoạt động ở chế độ đa nhiệm

- Tạo môi trường liên kết các đối tượng (OLE-Object Linking and Embeding)

- Hỗ trợ quản trị mạng.

Hệ điều hành Windows được phát triển theo hai hướng: Hệ điều hành máy tính cá nhân và hệ điều hành mạng. Trước khi đi sâu vào Hệ điều hành máy tính cá nhân chúng ta sẽ nói một chút về hệ điều hành mạng.

Hệ điều hành Windows NT:

Windows NT là hệ điều hành 32 bit, được thiết kế nhằm khai thác hết khả năng của các bộ vi xử lý như Intel x86, RISC và các hệ thống đa xử lý đối xứng (symmetric multiproccessing system).

Windows NT được thiết kế với giao diện tương tự như các phiên bản Windows khác tuy nhiên phần hạt nhân (kernel) đã được thiết kế khác nhằm bảo đảm cho Windows NT có thể thực hiện được hầu hết các chương trình đang chạy trên các họ máy tính x86 và RISC dưới MS-DOS, Windows, MS OS/2 version 1.x. Đặc biệt Windows NT đã đưa phát triển các chức năng bảo mật và quản trị nhằm đảm bảo tính an toàn cao cho các cơ sở dữ liệu và người sử dụng đầu cuối.

Hệ điều hành UNIX:

Hệ điều hành UNIX đã được Ken Thompson phát triển tại phòng thí nghiệm AT&T Bell, bang New Jersey từ năm 1969. Thương hiệu AT&T Bell là một trong những thương hiệu nổi tiếng thế giới trong lĩnh vực Công nghệ Thông tin . Từ năm 1969 đến nay UNIX đã trải qua một quá trình phát triển và ngày càng hoàn thiện.

Với chức năng là một hệ điều hành đa nhiệm, UNIX đã có những tính năng vượt trội so với các hệ thống đơn nhiệm cũ

* Cho phép nhiều người có thể sử dụng cùng một máy tính hoặc nhiều chương trình cùng xử lý một lúc (đa nhiệm).

* Cho phép từng cá nhân có thể thông tin trực tiếp với các máy tính khác thông qua hệ thống mạng và thiết bị đầu cuối .

* Tạo điều kiện dễ dàng trong việc chia xẻ tài nguyên và chương trình giữa các cá nhân với nhau.

* Với tính năng mở UNIX được nhiều hãng phần mềm phát triển và ngày càng trở nên phổ cập trong ngành Công nghệ Thông tin .

II. CÁC THÀNH PHẦN CƠ BẢN CỦA HỆ ĐIỀU HÀNH WINDOWS

1. Cài đặt hệ điều hành

Để cài đặt hệ điều hành Windows bạn cần mua một đĩa cài đặt, hiện nay đĩa cài đặt có bản quyền của hãng MicroSoft có giá vào khoảng 400 USD. Theo thỏa thuận của hãng với Chính phủ Việt Nam, đĩa cài đặt dành riêng cho ngành Giáo dục có giá khoảng 180 USD.

Sau khi lắp đĩa vào ổ CD bạn chỉ cần làm theo các chỉ dẫn trên màn hình, việc cài đặt diễn ra hầu như tự động.

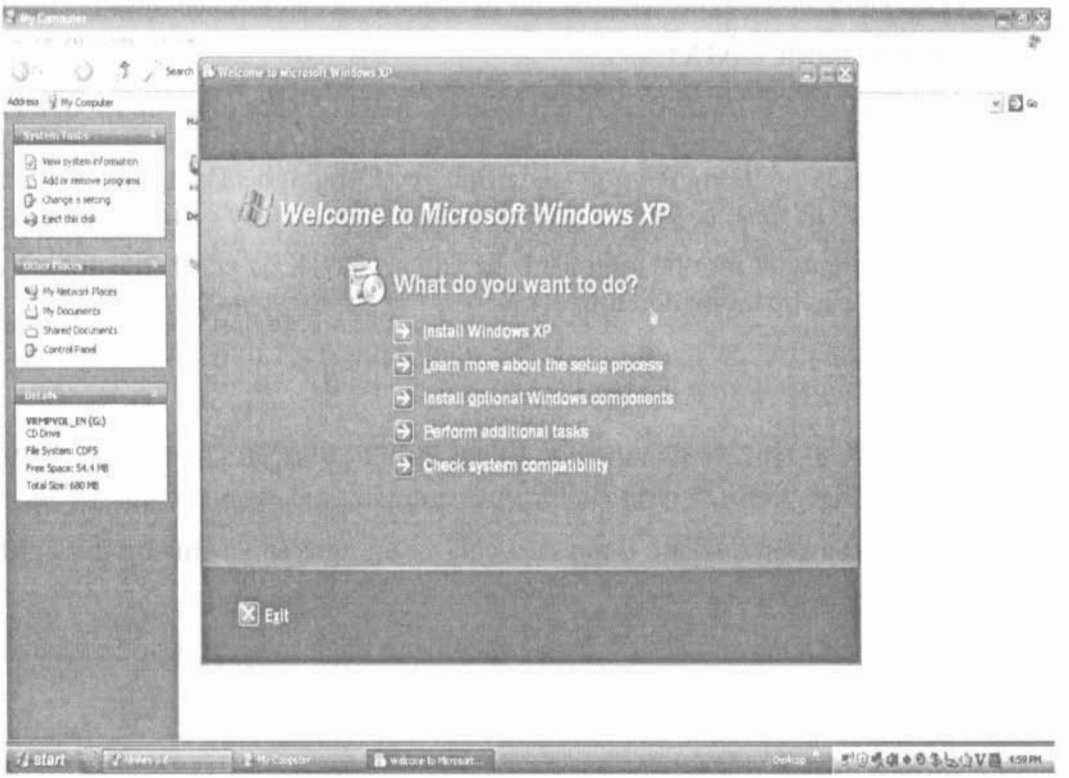
Trên màn hình máy sẽ đưa ra các lựa chọn (hình 4.1), để cài đặt hệ điều hành bạn chọn mục:

Install Windows XP,

trên màn hình kế tiếp chọn mục:

New Installation (hình 4.2)

Các bước tiếp theo bạn phải lựa chọn cài đặt tự động hay tự chọn các thành phần của Windows. Thời gian cài đặt là tương đối lâu vì vậy trên màm hình xuất hiện lời khuyên "Bạn hãy ngồi xuống và thư giãn trong khi máy làm việc"



Hình 4.1

Kết thúc cài đặt bạn phải khởi động lại máy để chương trình chép nốt các thành phần cần thiết trước khi bạn có thể làm việc.

Lời khuyên của chúng tôi là nếu bạn không giỏi tiếng Anh và không chuyên nghiệp thì nên tìm các chuyên gia, không nên tự làm.

Màn hình nền (Desktop) của Windows XP có hai dạng, dạng cổ điển và dạng hiện đại. Nếu bạn quen với màn hình cổ điển (Windows 2000) thì có thể chọn dạng đó bằng cách: đưa chuột xuống vị trí trống ở dòng cuối (đáy màn hình). Bấm phím phải và chọn Properties – chọn phiếu Start Menu – chọn tiếp Classic Start Menu (hình 4.3).

Hình 4.2

Hình 4.3

2. Khởi động hệ điều hành

Hệ điều hành được cài đặt trong ổ C của máy vi tính. Sau khi bật điện (ấn nút Power) hệ điều hành được nạp vào bộ nhớ Ram, kết thúc quá trình khởi động màn hình chính của Windows (Desk top) như sau (hình 4.4).

Windows đã có sẵn một số hình ảnh để bạn chọn làm nền, nếu không thích bạn có thể chụp các bức ảnh và đưa nó làm màn hình nền.

Hình 4.4

3. Các thành phần của màn hình nền

Màn hình nền có hai vùng chính, vùng hiện các biểu tượng và thanh tác vụ Taskbar, thanh này ngầm định nằm ở đáy màn hình tuy nhiên có thể di chuyển nó đến vị trí khác.

3.1 Màn hình chính (hình 4.4)

Có ba loại biểu tượng trên màn hình chính của Windows:

- Các thành phần của hệ điều hành: My Computer, Recycle Bin, … bạn có thể đổi tên các biểu tượng này bằng cách đưa chuột chỉ vào biểu tượng, bấm phím phải rồi chọn Rename. Tuy nhiên đây là điều bạn không nên làm.

- Các Shortcut: biểu tượng của các trình ứng dụng cài đặt trong máy, loại biểu tượng này có một mũi tên màu đen ở góc dưới bên trái. Khi muốn mở 1 trình ứng dụng ta chỉ việc nháy kép chuột vào biểu tượng tương ứng.

- Các Folder (thư mục) hoặc các tệp người sử dụng đặt trên Desktop

3.2 Thanh tác vụ Taskbar

Đáy màn hình là thanh tác vụ (thanh ứng dụng) TASKBAR (hình 4.5)

Hình 4.5

Phía ngoài cùng bên trái thanh Taskbar là nút Start tiếp theo là biểu tượng các ứng dụng đã kích hoạt. Tất cả các cửa sổ ứng dụng khi đưa về chế độ cực tiểu đều xuất hiện trên thanh Taskbar. Để cửa sổ hiện lại bình thường cần nháy kép vào biểu tượng của nó trên thanh này.

Sử dụng thanh Taskbar

Bấm phím phải chuột lên nền của Taskbar sẽ xuất hiện hộp thoại.

Các mục chọn trong hộp chọn có ý nghĩa như sau:

* Toolbars: Các thanh công cụ, mục chọn này có các lựa chọn

- Desktop: đưa các biểu tượng trên màn hình chính của Windows lên thanh Taskbar.

- Cascade Windows: Các cửa sổ đang mở xếp phủ lên nhau kiểu mái ngói.

- Tile Windows Horizontally: Các cửa sổ ứng dụng xếp phủ lên nhau theo chiều ngang.

- Tile Windows Vertically: Các cửa sổ ứng dụng xếp cạnh nhau theo chiều đứng.

- Show the Desktop / Show open Windows : đóng các cửa sổ, hiện thanh Taskbar hoặc hiện trở lại cửa sổ vừa bị đóng.

- Task Manager: quản lý các trình ứng dụng đang mở

- Lock Taskbar: khóa thanh tác vụ

- Properties: quy định cách thể hiện của Taskbar và thay đổi các thành phần

* Start Menu Programs.

Khi chọn mục này, hộp thoại Taskbar Properties xuất hiện. Các mục chọn trong hộp thoại này có các chức năng như sau:

- Always on top: Taskbar luôn xuất hiện trên màn hình ở tất cả các ứng dụng.

- Auto hide: Taskbar bị tự động dấu đi cho đến khi con trỏ chuột di chuyển đến vị trí của nó.

- Show small icon in Start menu: hiện các biểu tượng dưới dạng thu nhỏ trên bảng chọn Start.

- Show Clock: Hiện hay ẩn đồng hồ trên Taskbar.

3.3 Nút Start

Nút Start thực hiện nhiều chức năng quan trọng: tắt máy, chạy các ứng dụng, mở tệp, gọi cửa sổ trợ giúp, tìm tệp, ... và các công cụ thiết lập cấu hình khác.

Khi bạn nháy chuột ở nút Start, trên màn hình xuất hiện bảng chọn Start Menu Programs. Tùy theo việc cài đặt mà trong bảng chọn Start sẽ xuất hiện các mục khác nhau (hình 4.6).

Các mục chọn chính trong bảng chọn này có chức năng như sau:

Hình 4.6

- **All Programs**: hiện các chương trình ứng dụng trên Windows.
- **Command Prompt**: trở về cửa sổ của hệ điều hành DOS
- **Notepad**: khởi động trình soạn thảo Notepad
- MicroSoft Word: trình soạn thảo văn bản Word
- MicroSoft Office Excel: bảng tính điện tử
- E-mail: thư điện tử
- Internet: mạng toàn cầu
- Run: Chạy các chương trình.

- Search: tìm kiếm tệp hoặc thư mục

- Log off: thoát khỏi mạng

- Turn off computer: tắt máy

Khi chọn chức năng tắt máy, màn hình xuất hiện ba khả năng

Stand by	Khởi động lại Windows và vào chế độ MS – DOS.
Turn of	Tắt máy
Restart	Khởi động lại hệ điều hành Windows.

3.4. Chạy các trình ứng dụng

Việc chạy một trình ứng dụng phụ thuộc vào việc nó được cài đặt như thế nào. Nếu trình ứng dụng đã được đưa lên Desktop dưới dạng một biểu tượng ta có thể cho chạy trình này bằng cách bấm kép vào biểu tượng đó.

Nếu trên Desktop chưa có biểu tượng có thể chọn nút START/ chọn mục PROGRAMS/ chọn trình ứng dụng.

Trường hợp cuối cùng nếu hai cách trên không thực hiện được thì chọn nút START/ chọn mục RUN / chọn BROWSE/ chọn thư mục và trình ứng dụng/ chọn OK.

Cần chú ý rằng mỗi trình ứng dụng được lưu trong một thư mục (Folder) trên đĩa cứng, tệp chạy chương trình luôn có đuôi là .EXE hoặc .COM, tuy nhiên có rất nhiều tệp có đuôi như vậy nên cần biết tệp nào là tệp chạy chương trình .

Chú ý:

Trong quá trình làm việc máy bị treo (không hoạt động tiếp), để khởi động lại máy ta ấn nút RESET (hoặc ấn đồng thời 3 phím CTRL, ALT, DEL).

4. Tạo mới, xóa, thay đổi các biểu tượng hoặc tùy chọn trong một hộp thoại

4.1. Tạo biểu tượng hoặc mục chọn cho các ứng dụng

Đối với các chương trình độc lập ví dụ Pascal, Foxpro, Autocad ... ta có thể tạo một mục chọn nằm trong một bảng chọn nào đó hoặc tạo một biểu tượng trên Desktop để khi chọn mục chọn hoặc biểu tượng tương ứng thì chương trình thực hiện mà không phải làm thêm một động tác nào.

* Tạo biểu tượng trên Desktop

Bấm phím phải chuột – Chọn New – chọn Shortcut – chọn Browse cửa sổ Browse hiện như trong hình 4.7.

Đến đây bạn cần biết trình ứng dụng cài đặt trong thư mục nào, trên đĩa cứng nào (nếu máy của bạn có hơn 1 đĩa cứng, hoặc đĩa cứng được chia thành nhiều đĩa lôgic). Tìm đến nơi lưu trình ứng dụng và bấm đơn vào tên tệp chạy chương trình. Chọn OK sẽ xuất hiện hai khả năng:

- Trình ứng dụng đã chuẩn bị sẵn một biểu tượng để trên Desktop

- Bạn phải tự chọn một biểu tượng trong số các biểu tượng mà Windows đã thiết kế sẵn.

Sau các thao tác này, bấm Finish để kết thúc.

Chú ý:

Tất cả các trình ứng dụng chạy dưới DOS thường không tự đóng cửa sổ khi kết thúc và màn hình làm việc không phủ kín màn hình. Để khắc phục nhược điểm này bạn còn cần làm thêm các thao tác sau đây:

- Bấm phím phải chuột vào biểu tượng, chọn Properties, chọn Screen đánh dấu vào vòng tròn bên trái Full Screen. Tiếp đó chọn Program, đánh dấu vào hình vuông bên trái mục Close on Exit. Chọn Apply, chọn tiếp OK.

Hình 4.7

*** Tạo mục chọn trong một hộp thoại hoặc trong một Folder**

Việc này chỉ thực hiện được khi chúng ta sử dụng màn hình nền kiểu cũ (Classic Start Menu). Các thao tác tiến hành như sau:

- Bấm phím phải chuột trên thanh Taskbar, chọn Properties, chọn Start menu, chọn Classic Start menu để quay về màn hình kiểu cũ.

- Bấm nút Start.

Hình 4.8

- Chọn mục Settings, chọn Taskbar and Start menur, lúc này xuất hiện cửa sổ Taskbar and Start menu Properties, chọn Start menu, chọn Custome ta có cửa sổ mới (hình 4.8).

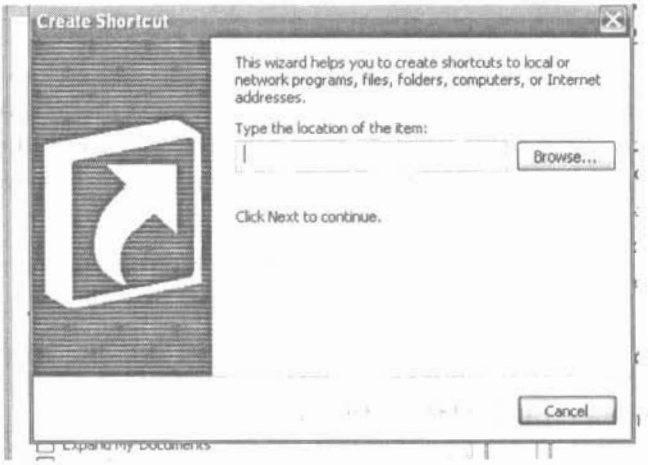
Các nút trong cửa sổ này gồm:

- Add: thêm biểu tượng mới

- Remove: xóa biểu tượng đã có

- Advanced: khai báo các phần mở rộng

- Sort: sắp xếp các biểu tượng



Hình 4.9

- Clear: xóa tất cả các lưu trữ đã có trong Recertly accessed document, program và Web sites

- Nhấn nút Add (để thêm biểu tượng) có cửa sổ hình 4.9

Dưới mục Type the location of the Item (hãy khai báo nơi chứa trình ứng dụng của bạn) có một khung chữ nhật trắng và một con trỏ nhấp nháy, nếu bạn biết trình ứng dụng nằm ở đâu thì khai báo đường dẫn và tệp chạy (đuôi .EXE) vào vị trí con trỏ nhấp nháy, nếu không nhớ chắc chắn thì thực hiện các thao tác sau:

- Nhấn nút Browse...

- Chọn thư mục chứa chương trình.

- Chọn tệp chương trình .

- Chọn Next. Khi này, cửa sổ Select Program Folder xuất hiện.

- Mục chọn Select a folder to place shortcut in (hãy chọn một vị trí để lưu trữ biểu tượng chương trình) Windows XP cho ta hai khả năng:

* Desktop

* Start menu

Nếu ta nhấn chuột trên Desktop, biểu tượng sẽ xuất hiện trên Desktop

Mục chọn Start Menu có một số lựa chọn sau

- Programs, biểu tượng sẽ xuất hiện tại trong khung Programs. Trong khung này bạn có thể đặt biểu tượng vào các Folder sau:

- Accessories

- Accessibility

- Entanrtainmemt

- StartUp

Nếu không muốn chọn bất kỳ Folder nào mà Windows đã khuyến cáo, bạn có thể tạo một Folder mới bằng cách bấm vào New Folder. Nếu chọn StartUp, chương trình sẽ được thực hiện ngay sau khi khởi động Windows. Nếu chọn New Folder, Windows sẽ tạo một bảng chọn mới trong bảng chọn Programs để chứa biểu tượng của chương trình.

Sau khi chọn xong nơi đặt biểu tượng, bấm Next để sang bước tiếp theo, lúc này xuất hiện cửa sổ Select a title for the Program (chọn tiêu đề cho chương trình).

Trong khung Type a name of this Folder (chọn cho biểu tượng của bạn một cái tên), ta cần đặt tên cho mục chọn này, nếu ta không đưa tên mới vào khung trên thì Windows sẽ đưa tên ngầm định

- Nhấn nút Finish kết thúc.

4.2 Thay đổi biểu tượng và tên của biểu tượng

a. *Thay đổi biểu tượng*

Để thay đổi biểu tượng cho một trình ứng dụng nào đó bạn có thể tiến hành như sau:

*** Tạo mục chọn trong một hộp thoại hoặc trong một Folder**

Việc này chỉ thực hiện được khi chúng ta sử dụng màn hình nền kiểu cũ (Classic Start Menu). Các thao tác tiến hành như sau:

- Bấm phím phải chuột trên thanh Taskbar, chọn Properties, chọn Start menu, chọn Classic Start menu để quay về màn hình kiểu cũ.

- Bấm nút Start.

Hình 4.8

- Chọn mục Settings, chọn Taskbar and Start menur, lúc này xuất hiện cửa sổ Taskbar and Start menu Properties, chọn Start menu, chọn Custome ta có cửa sổ mới (hình 4.8).

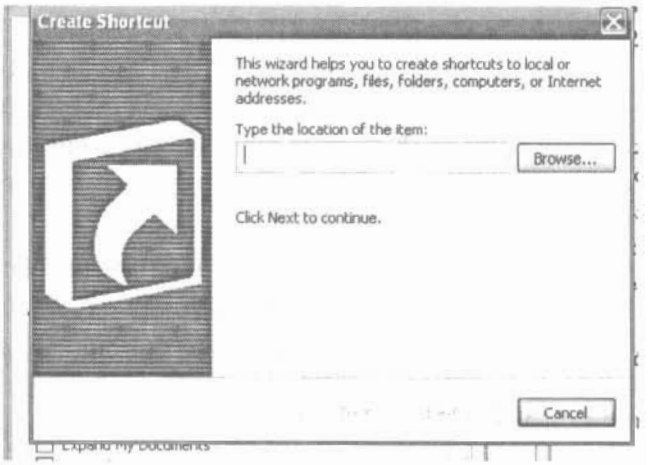
Các nút trong cửa sổ này gồm:

- Add: thêm biểu tượng mới

- Remove: xóa biểu tượng đã có

- Advanced: khai báo các phần mở rộng

- Sort: sắp xếp các biểu tượng



Hình 4.9

- Clear: xóa tất cả các lưu trữ đã có trong Recertly accessed document, program và Web sites

- Nhấn nút Add (để thêm biểu tượng) có cửa sổ hình 4.9

- Đưa con trỏ đến biểu tượng này

- Nhấn chuột phải, chọn Properties,

Chọn Program, chọn tiếp Change Icon để thay đổi biểu tượng. Khi này, cửa sổ Change Icon xuất hiện, trong cửa sổ này Windows đã thiết kế 114 biểu tượng, các biểu tượng này lưu trong tệp Moricons.dll. Bạn chỉ cần chọn biểu tượng cần thiết hoặc có thể nhấn nút Browse để tìm biểu tượng thích hợp trong các thư mục khác, chọn xong bấm OK.

Xin lưu ý: Có thể tự tạo biểu tượng bằng cách tạo hoặc sửa một hình ảnh và đặt tên tệp có phần mở rộng là *.ICO. Khi này, hình ảnh được tạo có thể làm biểu tượng cho một mục chọn nào đó.

b. ***Thay đổi tên biểu tượng***

Để thay đổi tên biểu tượng tiến hành như sau:

- Nếu biểu tượng nằm trên Desktop thì:

Bấm phím phải chuột vào biểu tượng xuất hiện hộp thoại, chọn Rename, xuất hiện một con trỏ nhấp nháy ở tên biểu tượng, gõ tên mới rồi OK.

- Nếu biểu tượng không nằm trên Desktop có thể tiến hành như dưới đây.

Vào mục chọn chứa tên trình ứng dụng (thường thì ở Program), chọn dòng chữ tên biểu tượng, bấm phím phải chuột, chọn Rename. Khi này, hộp thoại xuất hiện. Sửa lại tên biểu tượng (mục chọn) trong khung New name.

5. Sử dụng hộp thoại Control Panel

Hộp thoại Control Panel là hộp thoại rất quan trọng dùng để cài đặt các tham số liên quan đến giao diện của Windows, phần mềm ứng dụng, phần cứng của máy ...

Có nhiều cách để mở hộp thoại Control Panel:

Với màn hình kiểu truyền thống

- Bấm đơn vào biểu tượng Start, chọn mục Settings.

- Chọn Control Panel. Khi này, hộp thoại Control Panel xuất hiện (hình 4.10).

Nếu bạn đang dùng màn hình nền kiểu mới của Windows XP thì bấm phím chuột phải vào biểu tượng My Computer, chọn Open, chọn Control Panel hoặc bấm đơn vào nút Start, chọn Control Panel (xem hình 4.6)

Mỗi một biểu tượng trong cửa sổ Control Panel cho phép bạn thay đổi một số thông số nào đó của phần cứng cũng như cách thể hiện trên màn hình, trong phạm vi giáo trình này chúng ta chỉ có thể đề cập đến một vài chức năng chủ yếu.

Hình 4.10

5.1 Thay đổi tốc độ bấm phím và con trỏ màn hình

Trong hộp thoại Control Panel, bấm đơn phím phải chuột vào biểu tượng bàn phím (hình bên), chọn Open xuất hiện cửa sổ:


Keyboard

- Trong khung Repeat rate: Chọn tốc độ gõ các phím, kéo con trỏ đến vị trí Fast (nhanh) nếu bạn là người chuyên nghiệp có thể gõ 10 ngón với tốc độ cao, ngược lại bạn nên đặt thanh trượt gần với Slow (chậm).

- Trong khung Cursor blink rate: Thay đổi tốc độ nhấp nháy của con trỏ. Khi thanh trượt càng gần đến vị trí cuối (Fast) thì tốc độ nhấp nháy càng nhanh. Ngược lại, khi càng gần đến vị trí đầu (Slow) thì tốc độ càng chậm.


Display

5.2 Thiết lập các tham số cho màn hình

Trong hộp thoại Control Panel, bấm kép phím trái chuột vào biểu tượng có tên display (hình bên). Khi này, cửa sổ Display Properties xuất hiện (hình 4.11).

Ta chọn các mục sau:

**** Mục Themes***

Mục này cho phép tổ hợp màn hình nền, cài đặt âm thanh, cố định vị trí các biểu tượng và các thành phần khác. Mọi sự sắp xếp sẽ được lưu trữ khi bấm vào nút Save As.

Khi chọn khung này, hộp thoại Display Properties có dạng như hình trên.

Trong khung Wallpaper, chọn ảnh hoặc màu nền cho màn hình Windows. Có thể sử dụng nút Browse để chọn ảnh cho màn hình trong thư mục hoặc ổ đĩa khác.

Hình 4.11

**** Mục Desktop***

Mục này chọn ảnh nền cho Desktop sau khi khởi động Windows. Nếu bạn không thích những ảnh đã có trong thư viện của Windows bạn có thể đưa vào bất kỳ hình ảnh nào miễn là ảnh đó phải lưu trữ trong máy, để làm việc này chọn Browse, chọn đường dẫn tới nơi lưu trữ ảnh. Sau khi chọn xong ảnh bạn cần chọn cách thức mà bức ảnh hiện trên màn hình nền. Có ba cách hiện như sau:

- Stretch: hiện kín màn hình.
- Center: hiện thu nhỏ ở giữa màn hình.
- Tile: hiện nhiều hình trên cùng Desktop.

**** Mục Screen Saver* – Bảo vệ màn hình**

Cho phép chọn một hình ảnh, Windows XP đã thiết kế sẵn một thư viện ảnh và bạn chỉ cần chọn một ảnh ưng ý. Hình ảnh này sẽ xuất hiện sau một khoảng thời gian không sử dụng chuột hay bàn phím. Khoảng thời gian này tính bằng phút và được xác định trong khung Wait, có thể bấm vào đầu các mũi tên đen để tăng hoặc giảm thời gian đợi.

- Sau khi đã chọn một ảnh, nút Password protected phía bên phải có thể sử dụng để khai mật khẩu. Mật khẩu gõ trong khung New password và khai lại một lần nữa trong khung Confirm new password.

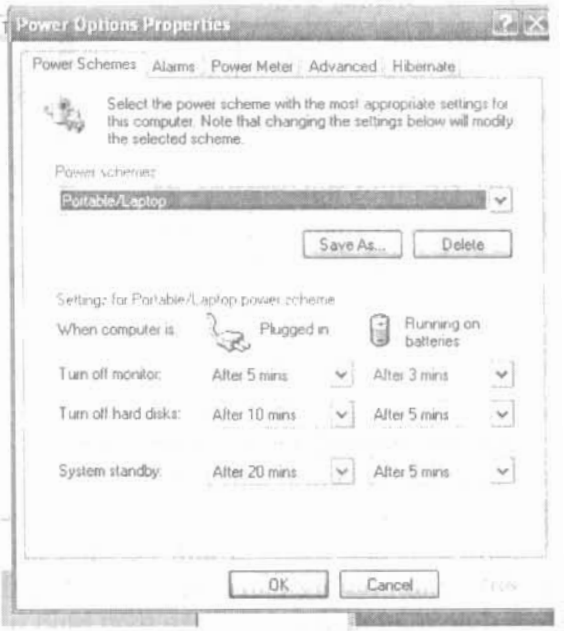
Khi đã thiết lập chế độ này, sau một khoảng thời gian xác định mà bạn đã lựa chọn nếu không có phím nào được bấm hay chuột không được di chuyển thì hình ảnh đã được chọn trong khung Screen Saver sẽ xuất hiện và cửa sổ hiện thời sẽ bị che đi. Nếu ấn một phím hoặc nhấn chuột thì hộp thoại xuất hiện. Bạn cần gõ đúng mật khẩu đã cài đặt để có thể làm việc với máy tính. Nếu mật khẩu đưa vào không đúng, ta không thể tiếp tục công việc.

- Khi cần xóa mật khẩu hoặc sửa đổi lại mật khẩu, bạn vào hộp thoại Change Password thực hiện tương tự như trên.

- Cửa sổ Screen Save còn có mục chọn Power. Bấm đơn vào mục chọn này ta có cửa sổ hình 4.12.

Mục Turn off monitor, xác định khoảng thời gian mà sau khoảng thời gian này nếu bạn không sử dụng đến máy tính (không nhấn các phím trên bàn phím hoặc nhấn chuột) thì màn hình sẽ tự động tắt.

Mục Turn off hard disks là tắt đĩa cứng. Nếu bạn dùng máy tính xách tay thì bạn cần lựa chọn khoảng thời gian cho chế độ sử dụng nguồn điện (Plugged in) hay sử dụng acquy (Running on batteries).



Hình 4.12

* ***Mục Appearance***

Mục này dùng để chọn sơ đồ màu cho màn hình nền, phông chữ, kích thước chữ cho các biểu tượng, các bảng chọn trong các ứng dụng của Windows …

* ***Mục Settings***

Mục này cho phép lựa chọn độ phân giải của màn hình (Screen resolution) và chất lượng màu.

Để thay đổi độ phân giải màn hình bạn chỉ cần kéo thanh trượt sang phải (tăng) hoặc sang trái (giảm).

Chất lượng màu thể hiện qua hai tham số:

- Highest (32 bit) chất lượng tốt nhất

- Medium (16 bit) chất lượng trung bình.

Chú ý:

1. Độ phân giải gồm hai thông số: Số điểm ảnh (Pixel) theo chiều ngang và theo chiều dọc màn hình. Các máy tính cá nhân hiện nay bao giờ số điểm ảnh theo chiều ngang cũng lớn hơn số điểm ảnh theo chiều dọc. Khi kéo thanh trượt có thể có lúc bạn vô tình đảo ngược giá trị của hai tham số này. Khi đó màn hình của bạn sẽ bị xoay đi và bạn phải ngoẹo đầu mới đọc được chữ, khi đó bạn cần khai báo trả lại các giá trị ban đầu.

Để kết thúc lựa chọn hãy bấm OK.

2. Thay cho việc chọn biểu tượng Display trong hộp Control Panel, có thể thao tác như sau: Bấm chuột phải vào một vùng trống trên Desktop , chọn Properties.

5.3 Thay đổi ngày tháng và thời gian cho hệ thống

Trong hộp thoại Control Panel, bấm kép vào biểu tượng Date and Time (hình bên), cửa sổ Date and Time Properties xuất hiện.

* Mục chọn Date and Time

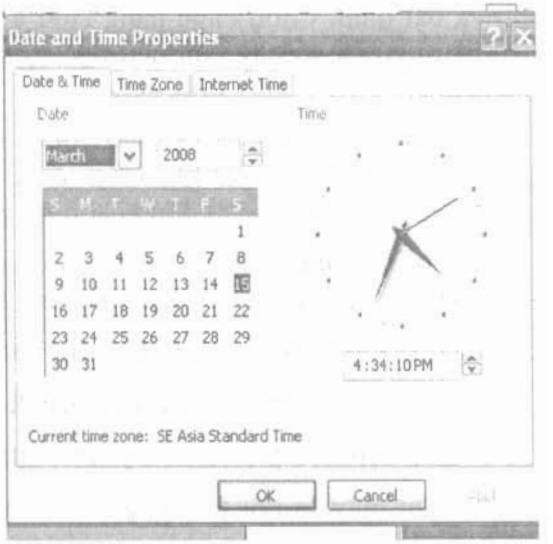
- Trong khung Date : Chọn ngày, tháng, năm cho hệ thống.

- Trong khung Time: Chọn giờ, phút, giây.

* Mục chọn Time zone: chọn múi giờ.

Hãy kéo thanh cuốn để chọn Bangkok, Hanoi, Jacacta là múi giờ của Việt Nam và một số nước Đông Nam Á khác (hình 4.13).

Sau khi đã thiết lập xong các tham số, nhấn nút Apply phía dưới hộp thoại để chấp nhận các giá trị.



Hình 4.13

5.4. Cài đặt máy in

Trước khi có thể sử dụng máy in cần tiến hành cài đặt trình điều khiển máy in.

Trong hộp thoại Control Panel, chọn biểu tượng printer and Faxes (hình bên). Khi này, cửa sổ Printers and Fax

xuất hiện (hình 4.14).

Hình 4.14

Để cài đặt máy in mới điều quan trọng là máy in phải được nối với máy tính và bật điện. Chọn chức năng Add a Printer, cửa sổ Add Printer Wizard xuất hiện. Trong cửa sổ này bạn nên đọc kỹ phần hướng dẫn của Windows tạm dịch như sau:

" Nếu máy in của bạn nối với cổng USB trên máy tính thì bạn không cần cài đặt máy in. Hãy thoát khỏi của sổ cài đặt, cắm đầu cáp máy in vào cổng USB và bật điện Windows sẽ tự động chọn máy in và cài đặt cho bạn".

Trường hợp máy in cắm vào cổng thông thường thì chọn Next để sang bước tiếp. Bước tiếp theo có hai khả năng lựa chọn

1. Local printer attached to this computer

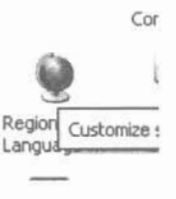
Automatically detect and install my plud and play printer

2. a netword printer or a printer attached to another computer

Nếu máy in nối trực tiếp vào máy tính của bạn thì chọn khả năng 1, còn nếu bạn cần in qua mạng hoặc có máy in dùng chung nhiều người và đang nối với máy tính khác thì chọn khả năng 2. Công việc còn lại máy sẽ làm tự động.

Sau khi cài đặt xong, máy sẽ hỏi bạn có muốn in thử một trang in không, bạn nên in thử để biết chất lượng mực in và máy in.

5.5 Chọn cách thức hiển thị dữ liệu



Để thiết lập chế độ hiển thị của dữ liệu (dạng số, tiền tệ, thời gian, ...), chúng ta cần tiến hành các bước như sau:

- Trong hộp thoại Control Panel, chọn biểu tượng regional and and language Options (hình bên).

Khi này, hộp thoại Reional and Langage Options xuất hiện (hình 4.15).

Các mục chọn gồm:

* Regional Options

* Languages

* Advanced

Hình 4.15

Trong mục chọn Regional Options có thể thay đổi cách thức hiển thị số (Number), tiền tệ (Curency), thời gian (Time) và cách hiện ngày tháng năm dạng rút gọn (short date) hoặc dạng đầy đủ (Long date).

Để làm việc này hãy bấm đơn vào nút Customize, xuất hiện cửa sổ hình 4.16

Nhấn chuột vào nút Number để chọn chế độ hiển thị cho các dữ liệu dạng số.
Lúc này phía dưới cửa sổ sẽ là các dòng cho phép lựa chọn cách thức hiển thị:

- Decimal symbol: chọn dấu phân cách phần nguyên và phần thập phân (có hai dấu là dấu phảy "," và dấu chấm "."). Theo ngầm định kiểu Mỹ là dấu chấm, còn nếu chọn kiểu Việt Nam thì dùng dấu phảy.

- Number of digits after decimal: số chữ số sau dấu ngăn cách thập phân

- Digits grouping symbol: dấu phân cách từng nhóm 3 chữ số, nếu dấu ở mục Decimal symbol là dấu chấm thì mục này chọn là dấu phảy và ngược lại.

- Digits grouping: chọn số chữ số trong mỗi nhóm

Hình 4.16

- Negative sign symbol: ký hiệu dấu âm

- Negative number format: Hiện số âm dưới một trong hai dạng sau:

  -123.456 hoặc (123.456)

- Display leading zero: hiện hoặc không hiện số 0 khi phần nguyên bằng 0

- List separator: chọn ký hiệu phân cách nhóm số

- Measurement system: hệ thống đo lường (US kiểu Mỹ).

Bạn có thể nhấn chuột lên khung Currency, Time và Date trong hộp thoại Customize Regional Options để xác lập chế độ hiển thị cho dữ liệu kiểu tiền tệ, thời gian và ngày tháng.

6. Trình tiện ích Windows Explorer

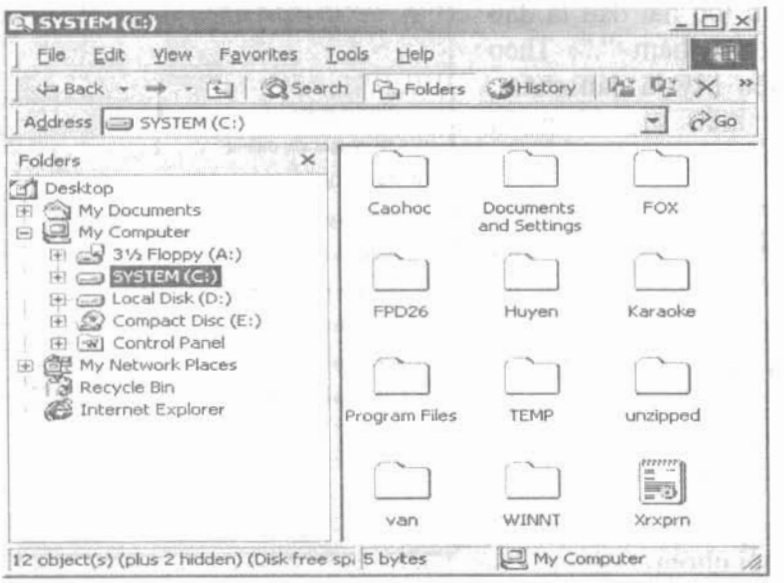
Windows Explorer là một trình tiện ích được thiết kế trong các phiên bản Windows 95/98/2000, ... dùng để sao chép, xóa, đổi tên các tệp, tạo các thư mục mới. Sang Windows XP trình này không còn xuất hiện độc lập và cũng không còn các tính năng như các phiên bản cũ nữa. Với các phiên bản cũ có thể khởi động Windows Explorer bằng một trong 3 cách sau:

- Cách 1: Nháy kép vào biểu tượng **Windows Explorer** trên Desktop

- Cách 2: Chọn nút **Start** / chọn mục **Program**/ chọn **Windows Explorer**

- Cách 3: Chọn nút **Start** / chọn mục **Run** / chọn nút **Browse** / chọn thư mục và chọn chương trình ứng dụng.

Cửa sổ làm việc của Windows Explorer có dạng như hình 4.17



Hình 4.17

Hình 4.18

Với Windows XP để tìm trình tiện ích Explorer cần thực hiện các thao tác sau:

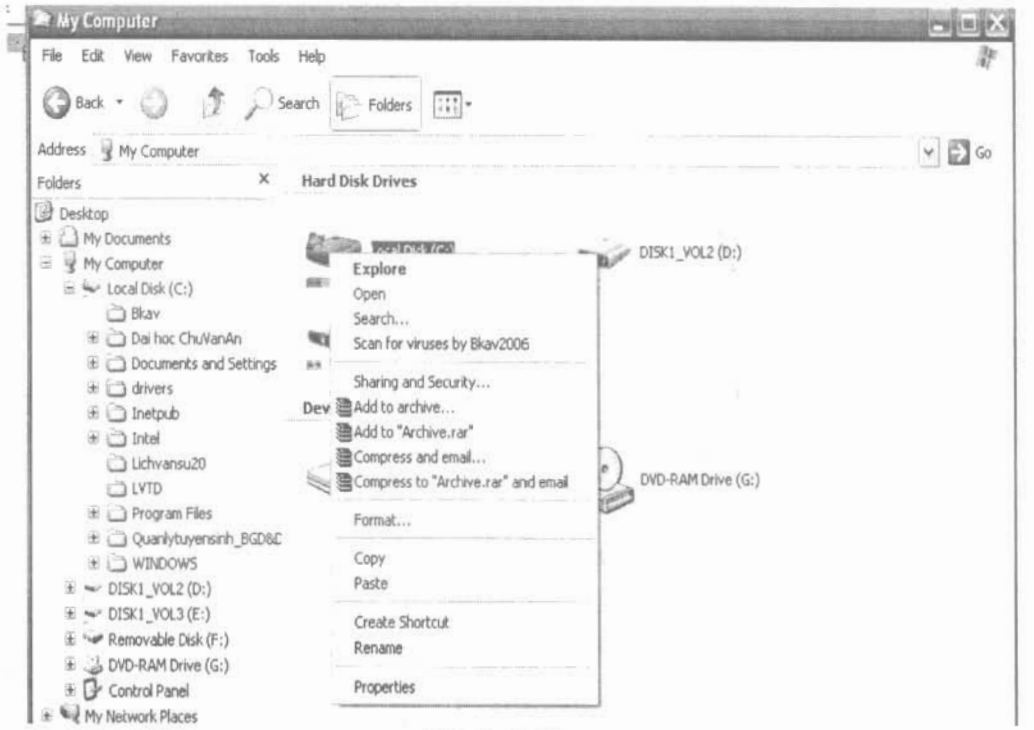
* Bấm đơn phím phải chuột vào biểu tượng My Coputer trên màn hình nền (Desktop), chọn Open ta có hình 4.18.

* Bấm đơn phím phải chuột vào biểu tượng một đĩa hoặc một thư mục bên phải hình 4.18 ta có hình 4.19. Tại đây bạn sẽ thấy chức năng Explorer.

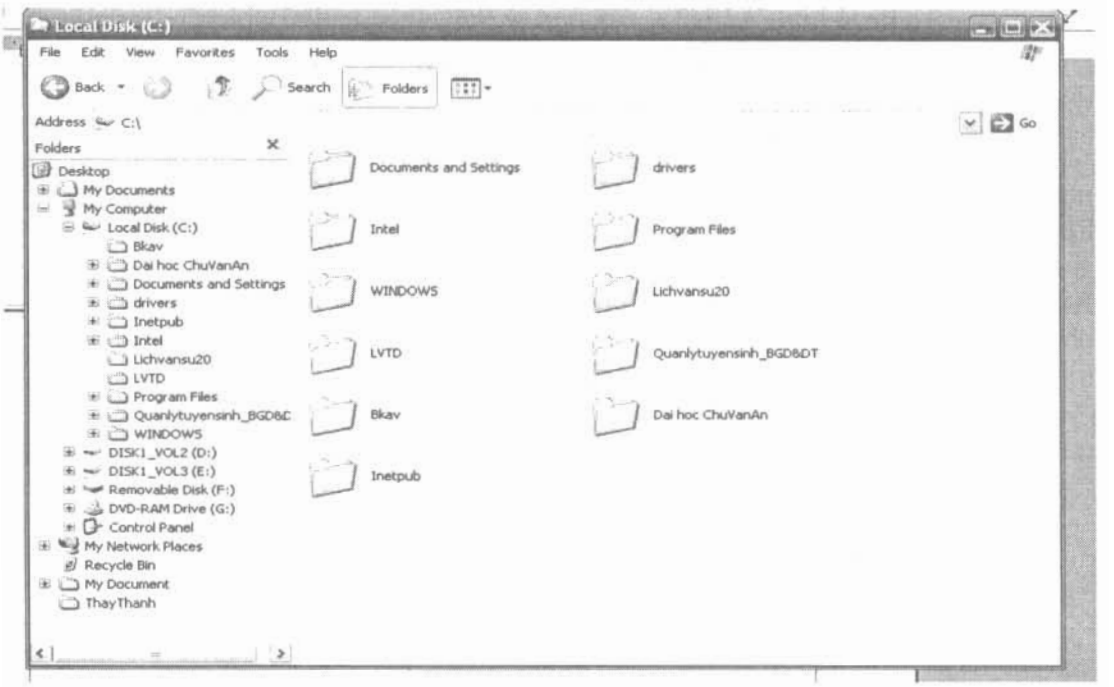
Cần lưu ý là mục chọn Explorer không phải lúc nào cũng nằm ổ đỉnh hộp thoại mà có thể năm ở dưới.

* Bấm đơn vào mục Explorer xuất hiện cửa sổ hình 4.20, cửa sổ này giống như cửa sổ trên hình 4.17 của các phiên bản Windows cũ nhưng các tính năng thì khác.

Cửa sổ làm việc của Windows Explorer được chia thành hai cửa sổ con: Khi đưa con trỏ đến một thư mục hay ổ đĩa ở cửa sổ bên trái thì trên cửa sổ bên phải sẽ xuất hiện nội dung của thư mục hay ổ đĩa này. Sau đây chúng ta sẽ tìm hiểu một số tính năng mà Explorer cung cấp:



Hình 4.19

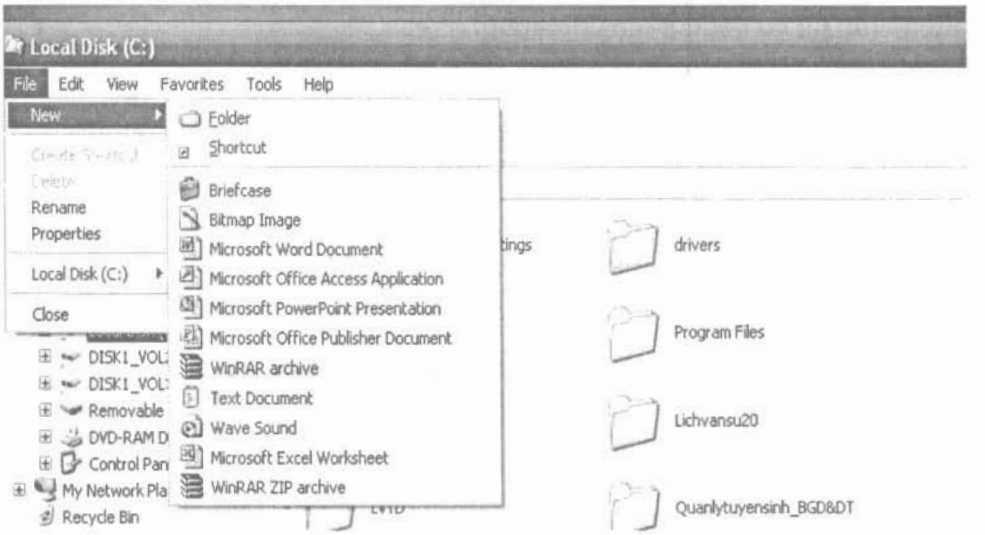


Hình 4.20

6.1. Tạo Folder hoặc Shortcut mới

- Trên cửa sổ bên trái, bấm đơn vào ổ đĩa hoặc thư mục đã có để biểu tượng nằm trên nền xanh. Chọn File, chọn New xuất hiện hộp thoại (hình 4.21).

- Chọn Folder để tạo thư mục mới. Khi này, trên cửa sổ bên phải xuất hiện một biểu tượng như hình bên. Biểu tượng được tạm đặt tên là New Folder. Có một con trỏ đang nằm ở New Folder, bạn có thể chọn một tên mới cho biểu tượng vừa tạo ra.



Hình 4.21

- Chọn Shortcut để tạo biểu tượng thực thi một trình ứng dụng hoặc một công việc nào đó. Khi chọn Shortcut chúng ta lại có một cửa sổ như hình 4.9. Chọn Browse để khai đường dẫn tới trình ứng dụng mà bạn muốn.

Chú ý:

Chức năng New trong thực đơn File nói ở trên chỉ xuất hiện khi bạn chọn một biểu tượng phía bên trái. Nếu chọn biểu tượng phía bên phải thì sẽ không thấy chức năng này.

6.2 Sao chép, di chuyển, xóa, đổi tên tệp hoặc thư mục

Giả sử chúng ta quy ước với nhau rằng nói " chọn một đối tượng" nghĩa là bấm đơn vào biểu tượng cho nó nằn trên nền xanh. Chọn chức năng Edit trên thực đơn trên đỉnh cửa sổ hình 4.21 ta sẽ thấy một hộp thoại hiện lên. Trong hộp thoại này có một số mục chọn phục vụ việc sao chép, di chuyển ... là Cut (xóa và lưu tạm trong bộ nhớ), Copy (sao chép), Paste (chuyển), Delete (xóa đưa vào sọt rác), Rename (đổi tên)..

Để sao chép một tệp hoặc một thư mục bạn cần bấm đơn vào biểu tượng hoặc thư mục đó để nó nằm trên nền xanh.

Chọn Edit, trong hộp thoại chọn tiếp Copy,

Tìm ổ đĩa hoặc thư mục bạn muốn sao chép tới, bấm đơn vào biểu tượng rồi quay về Edit, chọn Paste.

Trong các thao tác đã nêu trên thay vì chọn Copy bạn chọn Cut thì đối tượng bạn đã chọn sẽ được chuyển tới vị trí mới.

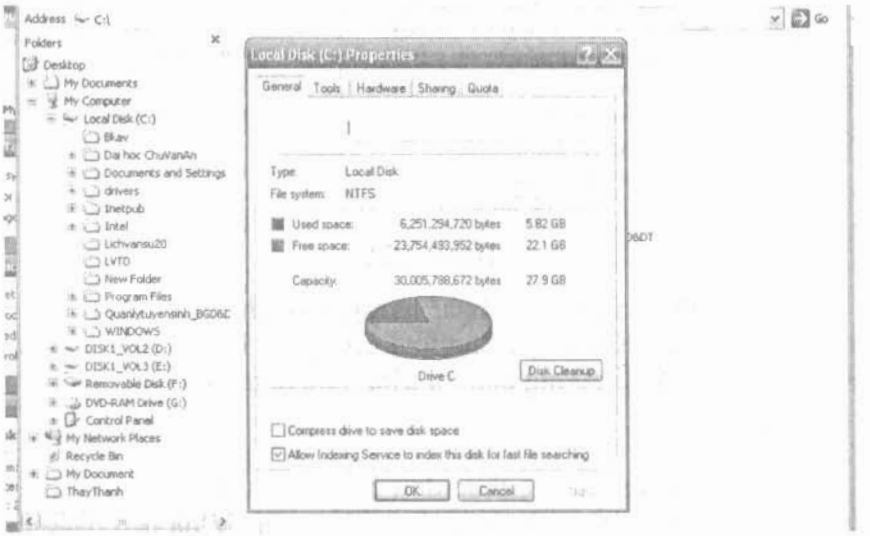
Sau khi đã chọn đối tượng nếu chọn Deletethì đối tượng sẽ bị xóa, còn nếu chọn Rename thì có thể đổi tên đối tượng.

Chú ý:

1. Các thao tác đã nêu có thể thay bằng cách bấm các tổ hợp phím:

Cut: Ctrl + X, Copy : Ctrl + C, Paste: Ctrl + V

2. Để chọn một nhóm các thư mục hoặc tệp để sao chép hoặc dịch chuyển, trước khi chọn tệp hoặc thư mục mới, nhấn và giữ phím Ctrl.



Hình 4.22

6.3 Xem thông tin

Có nhiều cách xem thông tin về một đối tượng (đĩa, thư mục, tệp). Nhìn vào cây thư mục phía bên trái (hình 4.21), bạn thấy một số đối tượng có một dấu cộng, bấm đơn vào dấu cộng này ta có thể xem được những gì có trong đối tượng đó. Các đối tượng bên trong lại có các dấu cộng bên trái và ta có thể tiếp tục tìm hiểu cho đến khi xuất hiện dấu trừ.

Nếu bấm vào dấu trừ thì bạn đã đóng đối tượng đó lại và quay về mức cao hơn.

Muốn xem những thông tin tổng quát như kiểu đối tượng (đĩa, thư mục, tệp), dung lượng mà đối tượng đang sử dụng, ngày tạo ra đối tượng ... Bạn cần bấm phím phải chuột trên đối tượng, chọn Properties, chọn General (hình 4.22).

Trên hình 4.22 chúng ta đang xem thông tin về đĩa C. Mục Type cho biết đây là đĩa cứng. Mục file system cho biết hệ thống đang dùng là NTFS. Mục Capacity cho biết tổng dung lượng đĩa. Mục Used space cho biết đã dùng hết bao nhiêu (màu xanh) và mục free space (màu tím) cho biết trên đĩa còn bao nhiêu dung lượng chưa dùng.

6.4 Tìm kiếm thông tin

Bấm đơn vào nút Start chọn Search ta có cửa sổ hình 4.23

Câu hỏi mà Windows đặt ra cho bạn là: What do your want to search for? Bạn muốn tìm kiếm cái gì?

Windows XP cho ta một số phương án trả lời:

- Pictures, music, or video: tìm các tệp ảnh, tệp nhạc hoặc tệp video

- Document: các tài liệu (có thể là văn bản, hoặc các trang tính..)

- All file and folders: tìm các tệp hoặc thư mục

- Computer or people: tìm các máy tính hoặc tìm người.

Nếu muốn tìm một văn bản đã lưu trữ trong máy hãy chọn All file and folders bạn sẽ thấy xuất hiện cửa sổ mới hình 4.24.

Hình 4.23

- Trong khung All or part of the file name: (tên đầy đủ hoặc một phần tên đối tượng) chọn tên tệp cần tìm, có thể sử dụng ký hiệu dấu sao (*) thay thế một nhóm ký tự bất kỳ và dấu hỏi (?) cho một ký tự bất kỳ.

- Trong khung A word or phrase in the file (một từ hoặc một đoạn nội dung có trong đối tượng), hãy gõ đoạn văn bản mà bạn nhớ có chứa trong tệp cần tìm.

- Trong khung Look in, chọn tên ổ đĩa hoặc thư mục cần tìm kiếm.

- Để việc tìm kiếm nhanh hơn bạn nên trả lời ba câu hỏi ở phía dưới:

* When was it modified: Tệp đã được sửa chữa khi nào? Bấm đơn vào biểu tượng hai mũi tên bên phải ta sẽ thấy các phương án trả lời (hình 4.25).

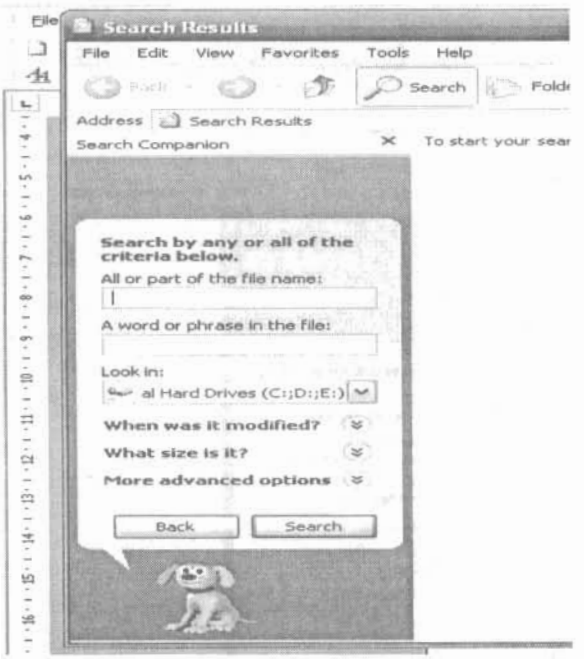
⊙ Don't remember (không nhớ)

○ Within the last week (tuần vừa qua)

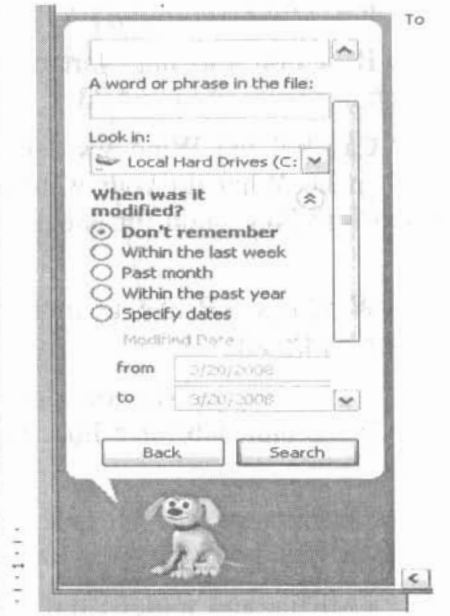
○ Past month (tháng vừa qua)

○ Within the past year (năm trước)

○ Specify dates (tự khai ngày tháng)



Hình 4.24



Hình 4.25

Nếu không nhớ rõ tốt nhất bạn nên chọn phương án 1 Don't remember (tôi không nhớ).

* What size is it : kích thước của đối tượng (tính theo Byte)

* More advanced options: một số phương án khác.

Sau khi khai báo xong các tham số trên, nhấn chuột lên nút Search để bắt đầu tìm kiếm.

Trong khung Search Results sẽ xuất hiện kết quả tìm kiếm.

7. Thay đổi chủ sở hữu tài nguyên

Bấm kép vào biểu tượng (hình bên) trong cửa sổ Control Panel để thay đổi (thêm, bớt) những người được quyền khai thác tài nguyên trên máy tính. Trên hình 4.26 bạn sẽ thấy ba yêu cầu mà Windows đặt ra khi muốn thực hiện công việc này:

- Change a account
- Create a new account
- Change the way users log on or log off

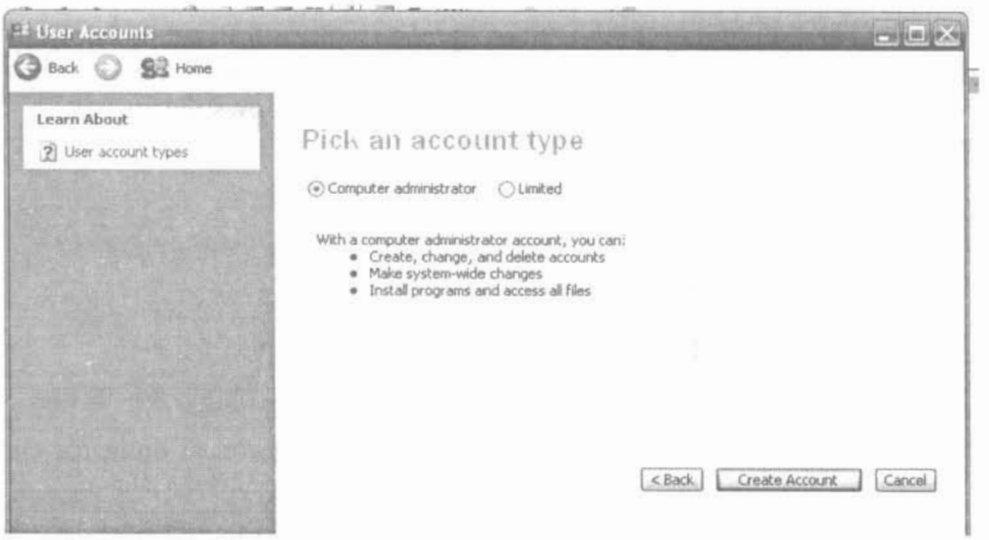
Để khai báo một chủ sở hữu tài nguyên mới hãy chọn mục:

Create a new account.

Trong cửa sổ mới xuất hiện bạn hãy khai báo tên người sử dụng rồi chọn Next

Có hai loại chủ sở hữu tài nguyên trên máy tính mà bạn cần lựa chọn (hình 4.26).

Hình 4.26.



Hình 4.27

Loại chủ sở hữu thứ nhất là **Computer aministrator:**

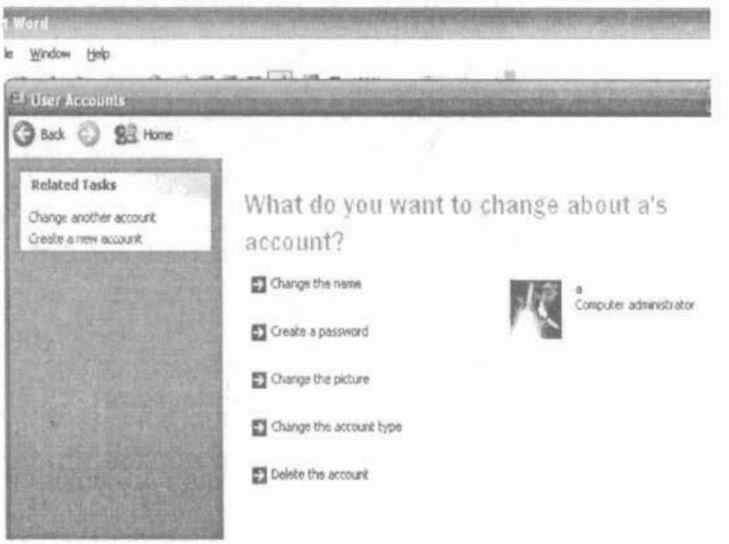
Đây là chủ sở hữu nắm quyền điều hành, là người có quyền cao nhất. Người này có thể cho phép tạo ra các chủ sở hữu mới, thay đổi hoặc xóa các chủ sở hữu khác. Cũng chỉ chủ sở hữu này mới có quyền cài đặt chương trình, truy cập vào bất kỳ tài nguyên nào hiện hữu trên máy.

Loại chủ sở hữu thứ hai là **Limited**

Những người thuộc loại này được phép thực hiện các thao tác sau:

- Tạo hoặc xóa mật khẩu của riêng mình,

- Thay đổi các thành phần của màn hình nền.



Hình 4.28

- Truy cập các tệp do mình tạo ra

- Truy cập vào các tệp được phép chia sẻ thông tin.

Giả sử bạn đã chọn

Create a new account:

- Trong khung Type a name for new account hãy chọn và gõ cho mình một cái tên. Chọn next để tiếp tục.

- Dưới mục chọn Pick an account type: Nếu bạn là chủ sở hữu máy tính bạn hãy chọn cho mình quyền cao nhất tức là: **Computer aministrator,** còn nếu bạn không phải là chủ thì chỉ nên chọn là **Limited.**

Hãy đánh dấu vào vòng tròn bên trái Limited rồi chọn Creat account. Sau khi chọn xong bạn sẽ thấy xuất hiện biểu tượng chủ sở hữu tài nguyên mới. Nếu muốn thay đổi các thông số của account này bạn hãy chọn biểu tượng account rồi chọn Change an account xuất hiện cửa sổ hình 4.28. Bạn có năm mục chọn để thay đổi các thông số:

- Change name: thay đổi tên account

- Create a password; tạo mật khẩu

- Change picture: thay đổi hình ảnh biểu tượng

- Change the account type: thay đổi loại (quyền) của chủ sở hữu

- Delete the account: xóa account

8. Định dạng đĩa (Format đĩa)

Sau khi đĩa đã được khởi tạo, tức là format bậc thấp rồi phân chia các đĩa logic, tức là các Partation bạn cần định dạng đĩa. Cần phải nói ngay rằng bạn phải hết sức thận trọng khi thực hiện thao tác này bởi vì toàn bộ dữ liệu và phần mềm cài đặt trên đĩa sau khi format sẽ bị xóa hết. Định dạng đĩa là công việc đòi hỏi bạn có một chút kiến thức chuyên nghiệp, đặc biệt là định dạng đĩa cứng. Tốt nhất là trước khi định dạng đĩa bạn nên chép hết những dữ liệu và nhừng phần mềm quan trọng sang đĩa khác hoặc sang USB. Giả sử cần format đĩa mềm (ổ A), hãy chọn biểu tượng My computer trên Desktop chọn đĩa A, nhấn nút chuột phải. Khi này, một hộp thoại xuất hiện. Chọn mục Format, chọn tiếp Start để bắt đầu định dạng. Bạn có thể chọn mục Quick Format để tiến hành Format nhanh.

PHẦN 2
NGÔN NGỮ PASCAL

Chương 1
MỞ ĐẦU

I. KHÁI NIỆM NGÔN NGỮ LẬP TRÌNH

Ngôn ngữ lập trình bao gồm một hệ thống các ký hiệu, các quy tắc cú pháp và câu lệnh dùng để viết thuật toán giải các bài toán. Tập hợp các câu lệnh hình thành các chương trình máy tính, chúng được lưu trữ dưới dạng các tệp. Những lệnh của chương trình sẽ được thực hiện lần lượt từ trên xuống (Top – Down) khi chúng ta ra lệnh chạy chương trình.

Những chiếc máy tính đầu tiên mà con người chế tạo ra ra các máy tính cơ học, khi đó ngôn ngữ lập trình chưa ra đời. Các thế hệ máy tính liên tục phát triển, do đó ngôn ngữ lập trình cũng phát triển theo. Có nhiều loại ngôn ngữ lập trình khác nhau, từ ngôn ngữ bậc thấp chuyển dần lên ngôn ngữ bậc cao.

* Ngôn ngữ bậc thấp (ngôn ngữ máy): Chỉ sử dụng hai chữ số 0 và 1 để mã hoá mọi đại lượng và phép toán. Ngôn ngữ này có ưu điểm là chương trình chạy nhanh, không phải biên dịch, tuy nhiên những chương trình viết bằng ngôn ngữ này rất cồng kềnh, viết mất rất nhiều thời gian, dễ sai sót, khó kiểm tra.

* Ngôn ngữ tập hợp: Trong ngôn ngữ này một số mã nhị phân được thay bằng các chữ cái do vậy việc lập trình đã dễ dàng hơn nhưng vẫn cần phải có chương trình dịch từ ngôn ngữ tập hợp ra ngôn ngữ máy.

* Ngôn ngữ bậc cao: Mô phỏng ngôn ngữ tự nhiên của con người do đó dễ dàng cho người lập trình, từ ngôn ngữ này phải qua một quá trình dịch sang ngôn ngữ máy do đó chương trình sẽ chạy chậm. Ngôn ngữ lập trình bậc cao đầu tiên ra đời năm 1958 và được đặt tên là ALGOL. Sau đó các nhà thiết kế ngôn ngữ đã giới thiệu nhiều ngôn ngữ khác như FORTRAN, BASIC, PASCAL, C++, JAWA, ...

II. TURBO PASCAL VERSION 7.0

Ngôn ngữ lập trình Pascal do giáo sư Niklaus Wirth - Trường Đại học Kỹ thuật Zurich, Thụy Sĩ thiết kế vào những năm đầu của thập kỷ 70. Ngôn ngữ được đặt tên là Pascal để tưởng niệm nhà toán học nổi tiếng người Pháp B. Pascal, người đã làm ra chiếc máy tính đầu tiên trên thế giới có bánh răng hiện số, cơ cấu bánh răng hiện số cho đến nay vẫn còn được sử dụng rộng rãi trong kỹ thuật.

Trước Pascal ngôn ngữ Fortran đã là một công cụ được sử dụng trong các trường học, tuy nhiên đây là một ngôn ngữ không có cấu trúc chặt chẽ nên việc lập trình gặp nhiều khó khăn, đặc biệt là khi cần bổ sung một đoạn chương trình mới vào một chương trình đã có. Hơn thế nữa với các máy tính điện tử lúc bấy giờ (thế hệ Minsk của các nước XHCN) chỉ với thao tác đục lỗ chương trình cũng đã có thể gây nhiều sai sót.

Với mục đích ban đầu là thiết kế một ngôn ngữ lập trình để giảng dạy cho sinh viên cách thức lập trình có cấu trúc, Pascal đã nhanh chóng trở thành một ngôn ngữ được tất cả những người làm tin học công nhận. Pascal đã trở thành công cụ kinh điển cho các nhà thiết kế ngôn ngữ sau này và nó cũng là ngôn ngữ được sử dụng trong các kỳ thi Tin học quốc tế.

Ngôn ngữ Pascal là một ngôn ngữ bậc cao có cấu trúc, trên cơ sở của ngôn ngữ Pascal chuẩn, hãng Borland đã đưa ra thị trường các phần mềm mang tên TURBO PASCAL. Turbo Pascal là một dạng biến thể của ngôn ngữ Pascal thông qua một quá trình biên dịch (Turbo) vì vậy làm việc với Turbo Pascal chúng ta sẽ không cảm thấy khó khăn như khi làm việc với Pascal chuẩn. Hiện nay các phiên bản TURBO PASCAL được sử dụng ở nước ta bao gồm từ version 3.0 đến 7.0. Những người đã thành thạo các version 3.0 hoặc 5.0 khi chuyển sang version 7.0 cũng không có gì khó khăn.

Ngôn ngữ Pascal có hai đặc điểm nổi bật là:

* PASCAL là ngôn ngữ lập trình có tính cấu trúc và tính hệ thống: các kiểu dữ liệu đa dạng, các cấu trúc điều khiển chặt chẽ, các cấu trúc khối trong chương trình rõ ràng...

* PASCAL là ngôn ngữ lập trình có định kiểu: các đại lượng (biến và hằng) đã được khai báo để sử dụng với kiểu dữ liệu này thì không thể đem dùng lẫn với kiểu dữ liệu khác.

III. CÀI ĐẶT PASCAL

Toàn bộ các tệp của Turbo Pascal 7.0 sau khi cài đặt chiếm vào khoảng 5 MB, tuy nhiên để có thể làm việc với Turbo Pascal kể cả phần đồ hoạ chúng ta có thể chép một số tệp cần thiết vào một đĩa 1,44 MB. Với khả năng đồ hoạ phong phú, Turbo Pascal cho phép chọn 16 màu trong các hình vẽ cho nên tốt nhất là làm việc với một máy màn hình SVGA hoặc TVGA. Nếu phải làm việc trên đĩa mềm thì nên dùng Pascal 5.0 và FORMAT đĩa thành đĩa hệ thống sau đó chép vào đĩa các tệp sau đây:

- TURBO.EXE: Chương trình soạn thảo, dịch, liên kết menu....

- TURBO.TPL: Tệp lưu các đơn vị chuẩn của Turbo Pascal

- GRAPH.TPU: Tệp lưu các đơn vị đồ hoạ chuẩn của Turbo Pascal

- ATT.BGI; CGA.BGI; EGAVGA.BGI; HERC.BGI; IBM8514.BGI; PC3270.BGI; VESA.BGI:

- BOLD.CHR; EURO.CHR; GOTH.CHR; SANS.CHR; SCRI.CHR .....

Các tệp *.BGI là các tệp chứa các điều khiển về kiểu màn hình, nếu ta dùng cố định trên một máy thì chỉ cần chép vào một tệp phù hợp với kiểu màn hình hiện có.

Các tệp *.CHR là các tệp chứa điều khiển kiểu chữ trong đồ hoạ, tuỳ theo dung lượng đĩa chúng ta có thể chép vào càng nhiều tệp càng tốt. Tuy nhiên cần lưu ý là phải dành chỗ để ghi các chương trình mà chúng ta sẽ tạo ra sau này, dung lượng để dành tối thiểu nên vào khoảng 200 KB.

Khi làm việc với Pascal có hai tệp đặc biệt được sinh ra để lưu giữ những quy định của người sử dụng và các thay đổi chương trình là TURBO.PCK và TURBO.TP, nếu không thật cần thiết thì không nên xoá các tệp này đi.

Với những máy tính cài đặt Window 98 hoặc các phiên bản cao hơn, sau khi cài đặt Pascal vào đĩa cứng và đã tạo ra Shortcut trên Desktop, nếu không khai báo gì thêm thì mỗi lần khởi động Pascal chương trình sẽ phải thoát hoàn toàn khỏi Window, khi kết thúc chương trình thì Window sẽ được khởi động lại. Để tránh khỏi điều phiền toái này chúng ta có thể xác lập lại một số thông số như sau:

Bấm phím phải chuột vào biểu tượng Turbo Pascal trên màn hình nền (Desktop), trong hộp thoại hiện lên trên màn hình bấm đơn phím trái chuột vào chức năng Properties chúng ta sẽ có cửa sổ sau (hình 0.1).

Trong cửa sổ hình 1 chọn chức năng Program, chọn tiếp Advanced ta có cửa sổ hình 0.2.

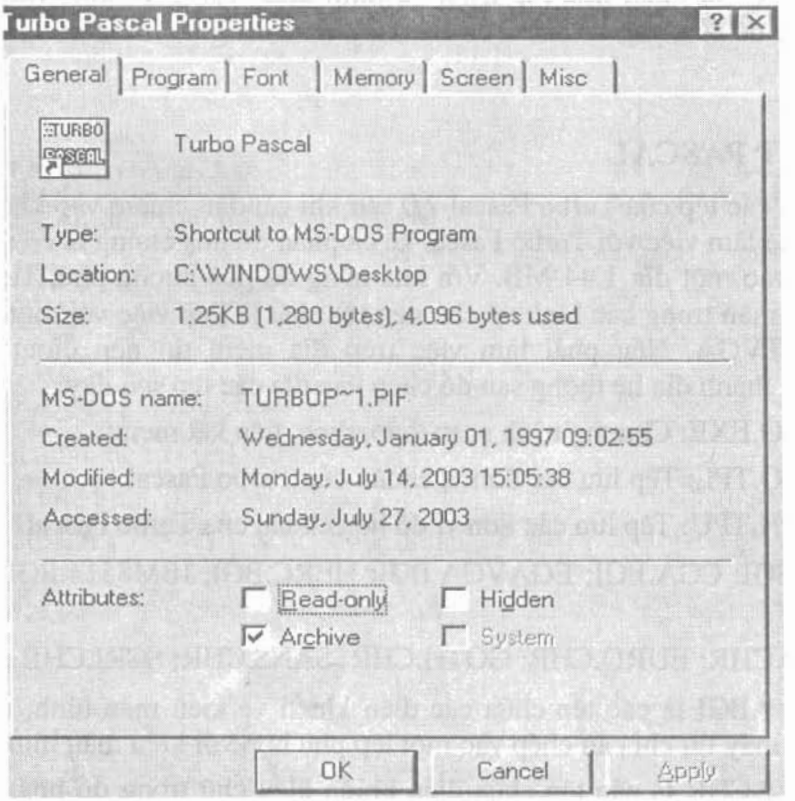
Trong cửa sổ hình 2 đánh dấu vào ô vuông bên trái các mục chọn:

Prevent MS-DOS based programs from detecting Windows

Sugget MS-DOS mode as necessary

Bấm OK hai lần để trở về Desktop.

Kể từ nay khi khởi động Pascal chúng ta vẫn không xoá bỏ hẳn hệ điều hành Window. Kết thúc làm việc với Pascal chúng ta lại quay về Desktop của Window.



Hình 0.1

IV. KHỞI ĐỘNG TURBO PASCAL

Tệp chương trình làm việc của Turbo Pascal là Turbo.exe, việc khởi động sẽ tuỳ thuộc vào sự cài đặt bộ chương trình trên đĩa của người sử dụng, giả thiết rằng tất cả các tệp đã nêu trên được chép vào thư mục TP của đĩa mềm đặt trên ổ A, lệnh khởi động Turbo Pascal sẽ là:

A:\>CD TP↵

A:\TP>TURBO↵

Nếu Pascal được cài đặt trong đĩa cứng và Shortcut đã có trên Desktop ta chỉ việc bấm kép vào biểu tượng này.

Nếu biểu tượng chương trình Pascal chưa có trên Desktop thì có thể tạo ra theo các bước sau:

* Bấm phím phải chuột, chọn New, chọn tiếp Shortcut.

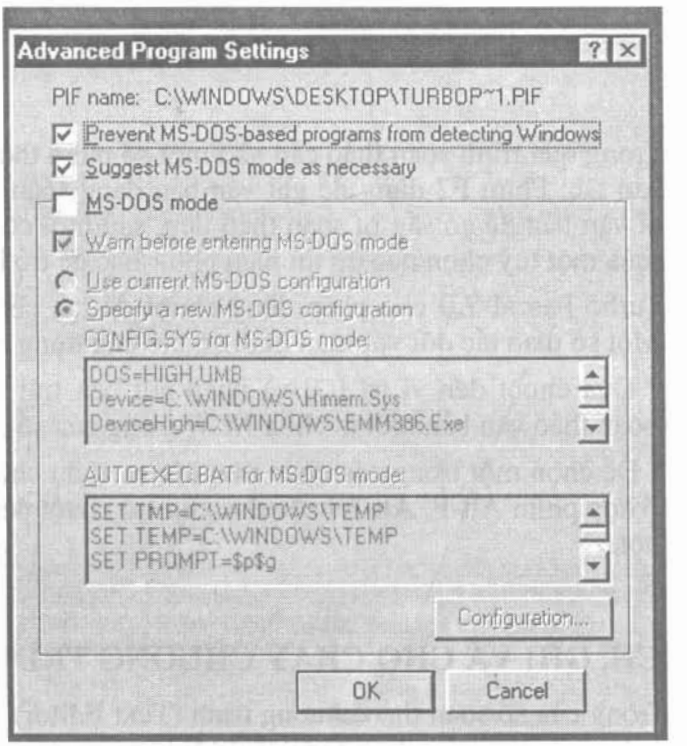
* Trong khung khai báo phía dưới mục **Command Line** gõ vào đường dẫn và tên tệp khởi động chương trình Pascal như sau:

C:\TP\BIN\TURBO

* Bấm vào Next để chuyển sang cửa sổ kế tiếp.

* Dưới mục **Select a name for the shortcut** ta có thể tuỳ ý chọn một tên cho biểu tượng sẽ hiện lên trên Desktop ví dụ TURBO PASCAL7.0

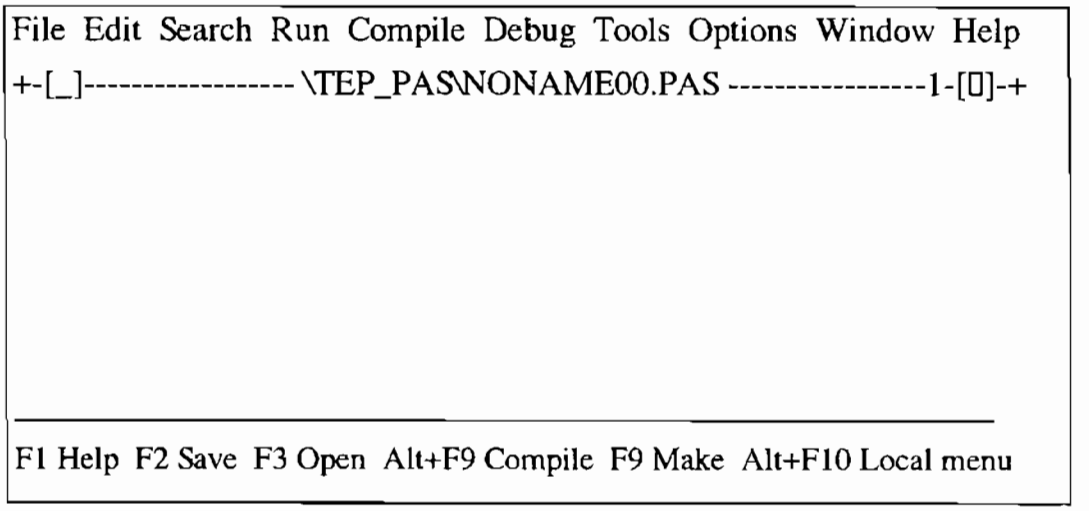
* Bấm vào Finish để kết thúc.



Hình 0.2

Sau khi khởi động sẽ xuất hiện màn hình làm việc của Turbo Pascal. Tuỳ thuộc vào version chúng ta sử dụng dòng thực đơn trên đỉnh màn hình có thể sẽ thay đổi chút ít, dưới đây là màn hình khi khởi động của Pascal Version 7.0.

Dòng trên cùng được gọi là thực đơn (MENU) chính của Turbo Pascal, mỗi chức năng trong Menu lại có các tuỳ chọn. Để kích hoạt một Menu nào đó ta bấm đồng thời phím Alt và chữ cái viết hoa trong tên của chức năng, ví dụ để sử dụng chức năng về tệp ta bấm và giữ phím Alt sau đó bấm tiếp phím F. Khi làm việc với các chức năng nếu gặp khó khăn có thể bấm phím F1 là phím trợ giúp để xem các hướng dẫn thao tác cần tiến hành.



Hình 0.3

Trong qua trình soạn thảo cần nhớ một số phím thông dụng để có thể rút ngắn thời gian thao tác: Phím F2 dùng để ghi văn bản đang soạn thảo vào đĩa, phím F3 dùng để mở một văn bản đã có sẵn ra soạn thảo tiếp, sau mỗi công đoạn nếu không muốn dùng cửa sổ của một tuỳ chọn nào đó thì bấm phím Esc để trở lại màn hình soạn thảo.

Turbo Pascal 7.0 cho phép dùng chuột để lựa chọn các chức năng hoặc các tuỳ chọn. Một số thao tác đối với màn hình có thể ứng dụng như sau:

* Đưa chuột đến vị trí [□] và nhấp nút bên trái sẽ xoá sạch toàn bộ màn hình. Muốn soạn thảo văn bản mới ta chọn NEW trong cửa sổ của chức năng FILE.

* Để chọn một trong các chức trên Menu ví dụ chức năng File hoặc Edit ta có thể dùng tổ hợp phím Alt-F, Alt-E hoặc đưa con trỏ chuột đến File hay Edit và nhấn nút bên trái chuột.

V. DỊCH, GHI VÀ CHO CHẠY CHƯƠNG TRÌNH

Trong cửa sổ soạn thảo chương trình (Text Editor) các chương trình mới soạn thảo lần đầu đều có tên ngầm định là NONAME00.PAS. Để ghi chương trình vào đĩa ta bấm phím F2, khi đó xuất hiện cửa sổ để ghi tên (hình 4).

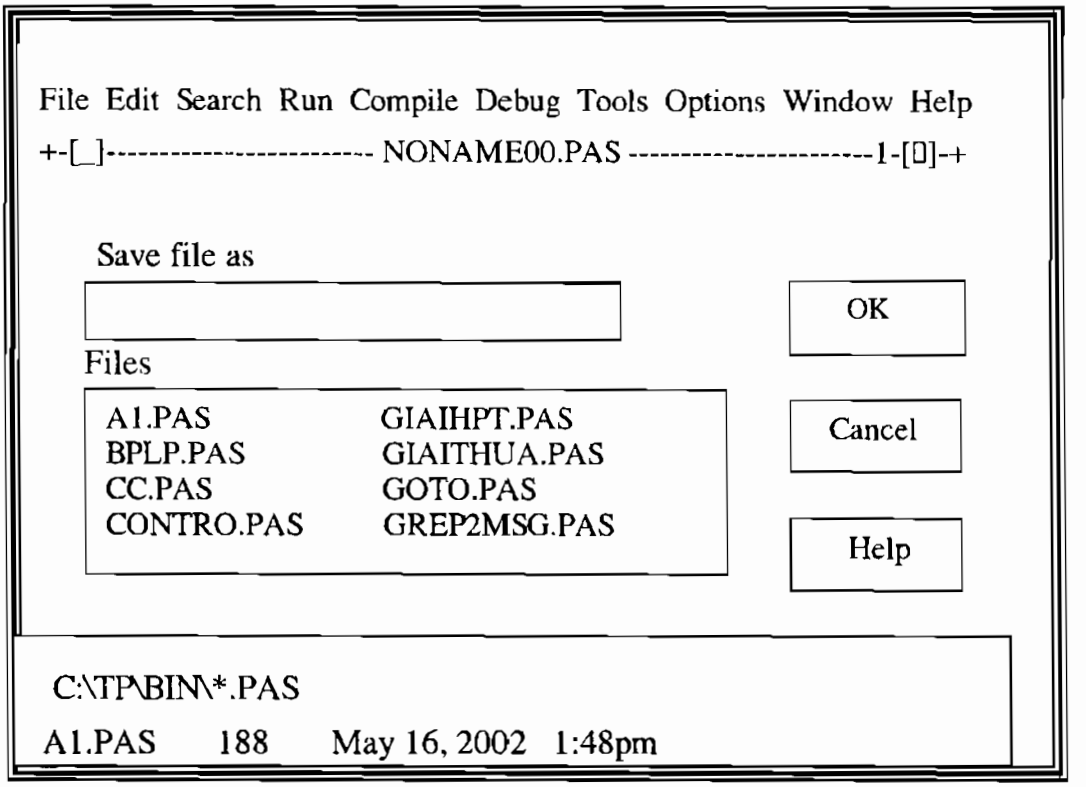
Sau khi soạn thảo xong một chương trình ta có thể cho chạy thử ngay bằng cách bấm tổ hợp phím Ctrl-F9. Nếu chương trình bị lỗi máy sẽ ra thông báo lỗi và ta phải gọi lại chương trình để sửa.

Dưới mục **Save File as** gõ tên tệp mà ta muốn chọn sau đó bấm OK chương trình sẽ được ghi vào đĩa với đuôi ngầm định của Pascal là PAS.

Chú ý

Khi tên tệp đã xuất hiện ở đỉnh cửa sổ thay cho Noname00.pas nếu chúng ta sửa chữa tệp và bấm F2 ghi lại thì Pascal sẽ không hỏi tên tệp nữa mà ghi lại mọi sự thay đổi vào tệp đã có.

Các tệp chương trình ghi vào đĩa với đuôi PAS là các tệp nguồn và đây là các tệp văn bản, các tệp này có thể in ra giấy bằng các lệnh TYPE hoặc COPY của DOS hoặc chuyển sang Word để in.



Hình 0.4

Trường hợp chương trình đã chạy tốt không bị lỗi. Ta có thể tiến hành dịch và lưu trữ các chương trình nguồn sang dạng chương trình có đuôi EXE, nghĩa là chuyển tệp dạng văn bản sang tệp thực hiện (Execute File). Những tệp đã dịch sang đuôi EXE thường có dung lượng lớn hơn tệp nguồn nhiều lần ví dụ một tệp nguồn đồ hoạ có dung lượng 1182 bytes khi dịch sang dạng duôi EXE sẽ có dung lượng 23120 bytes, vì vậy khi dịch cần lưu ý dung lượng còn được phép sử dụng trên đĩa. Những tệp đã dịch sau đó có thể cho chạy trực tiếp từ mức hệ điều hành bằng cách gọi trực tiếp tên tệp.

Để dịch chương trình ta bấm tổ hợp phím Alt-C, khi đó Menu cột của chức năng Compile (chức năng dịch) xuất hiện. Nếu bên cạnh tuỳ chọn Destination (nơi sẽ lưu tệp EXE) ta thấy có chữ Disk thì chỉ cần chọn Compile để tiến hành dịch, còn nếu là Memory (bộ nhớ) thì ta phải bấm D để chuyển thành Disk sau đó mới chọn Compile.

Tệp được biên dịch sẽ có cùng tên với tệp nguồn nhưng đuôi sẽ được đổi thành EXE, tệp này được ghi vào cùng thư mục với tệp nguồn. Cần lưu ý rằng sau quá trình biên dịch chúng ta sẽ có hai tệp cùng tên (một tệp có đuôi là PAS còn tệp kia là EXE) và không thể gọi tệp có đuôi EXE ra để sửa chữa. Nếu ta tiếp tục sửa chữa tệp nguồn thì phải tiến hành dịch lại sang đuôi EXE.

Chương 1

CÁC THÀNH PHẦN CƠ BẢN CỦA NGÔN NGỮ PASCAL

Con người thể hiện tư duy của mình bằng ngôn ngữ, khởi đầu là ngôn ngữ nói và sau đó là ngôn ngữ viết. Mỗi dân tộc dùng một số ký tự khác nhau cho ngôn ngữ viết của mình điều này cũng không ngoại lệ với các ngôn ngữ lập trình. Ngôn ngữ Pascal cũng có một bộ ký tự và những quy tắc riêng để hình thành nên các từ khoá, tên chuẩn và các câu lệnh của mình.

I. BỘ KÝ TỰ CỦA TURBO PASCAL

Bộ ký tự của Pascal chủ yếu giống như bộ ký tự ASCII, chúng gồm các nhóm sau:

- Nhóm 26 chữ cái in: A, B, C, ...Z,
- Nhóm 26 chữ cái thường: a, b, c, ...z
- Nhóm chữ số của hệ đếm thập phân: 0,1,2,3,4,5,6,7,8,9
- Nhóm toán tử toán học: +, -, *, /, >, <, =, >=, <=, <>
- Nhóm các ký tự đặc biệt: (,),[,],?,{, }...
- Dấu cách
- Dấu gạch nối _ (cần phân biệt với dấu trừ -)

Dấu gạch nối được dùng để phân cách các từ cho dễ đọc, Pascal không cho phép viết dấu trừ giữa các từ. Ví dụ:

giai_phuong_trinh_bac_hai; là cách viết đúng

giai-phuong-trinh-bac-hai; là cách viết sai.

Từ các ký tự, ngôn ngữ Pascal đã xây dựng một số từ khoá và tên chuẩn để sử dụng khi lập trình. Các từ khoá và tên chuẩn này bắt buộc phải dùng đúng cú pháp và không được phép sử dụng vào các công việc khác.

Các nhóm từ khoá của Pascal bao gồm:

**** Từ khoá chung:***

PROGRAM, BEGIN, END, PROCEDURE, FUNCTION

**** Từ khoá khai báo:***

CONST, VAR, LABLE, TYPE, ARRAY, RECORD, SET, FILE OF, STRING

**** Từ khoá dùng trong các cấu trúc lập trình***

* IF...THEN...ELSE...

* CASE.... OF...

* FOR...TO...DO...

* FOR...DOWNTO... DO...

* WHILE...DO...

* REPEAT...UNTIL...

* ***Từ khoá điều khiển***

* WITH, GOTO, EXIT

* ***Từ khoá tên các toán tử***

NOT, AND, OR, XOR, IN, DIV, MOD

Chú ý:

Pascal không phân biệt chữ in hay chữ thường, tất cả từ khoá hay câu lệnh đều có thể viết bằng chữ in, chữ thường hoặc xen kẽ chữ in và chữ thường.

II. QUY ĐỊNH ĐẶT TÊN

Tên là một dãy ký tự dùng để đặt cho hằng, biến, chương trình, nhãn hoặc kiểu dữ liệu mới... Tên được phép viết dài nhất là 127 ký tự nhưng thực tế chỉ có 63 ký tự đầu là có nghĩa. Tên được viết theo quy định sau:

- Tên phải bắt đầu bằng một chữ cái

- Không được dùng các ký tự đặc biệt

- Không được để khoảng trống giữa các ký tự

Pascal đã định nghĩa sẵn một số tên chuẩn như tên một số chương trình con, tên kiểu...

Boolean, char, integer, real, Byte, Text, read, readln, write,

writeln, abs, cos, sin, sqrt, exp, Ord, Round, Trunc, Pres, Succ...

Sự khác nhau giữa từ khoá và tên chuẩn là:

Người lập trình có thể định nghĩa lại tên chuẩn, dùng tên chuẩn vào các mục đích khác khi cần, còn từ khoá thì không được phép thay đổi và dù cố ý cũng không thể dùng từ khoá với những ý nghĩa khác với quy định mà Pascal đã thiết kế.

III. QUY ĐỊNH VỀ CÁC DẤU

Trong một câu lệnh các từ khoá phải viết cách nhau bởi dấu cách (bấm phím Space) còn tên hằng, tên biến phải viết cách nhau bởi dấu phảy ",". Giữa các câu lệnh phải dùng dấu " ; " để báo hết câu. Có thể viết nhiều câu lệnh trên cùng một dòng tuy nhiên nếu viết dài quá 127 ký tự thì Pascal sẽ thông báo lỗi:

Line too long : dòng quá dài

IV. PHÉP GÁN GIÁ TRỊ

Để gán giá trị cho hằng , biến ta phải dùng ký hiệu gán " := "

Ví dụ: a:= 3; Lam:='c';

V. LỜI GIẢI THÍCH

Trong chương trình những ghi chú hoặc lời giải thích có thể đưa vào trong cặp dấu {....} hay (*.....*). Nếu chúng ta muốn tạm thời bỏ một số câu lệnh trong chương trình thì cũng có thể dùng một trong hai nhóm ký tự trên , lưu ý là nếu ta đã mở đoạn ghi chú bằng '{' thì cũng phải đóng bằng '}' chứ không được dùng lẫn lộn hai loại ký tự. Lời giải thích hoặc ghi chú có thể đặt tại bất kỳ nơi nào trong chương trình ví dụ:

(* Chuong trinh tinh tien gui ngan hang *)

{ Chuong trinh tinh tien gui ngan hang }

VI. CẤU TRÚC CHƯƠNG TRÌNH

Một chương trình lập bằng ngôn ngữ Pascal thông thường phải gồm ba phần chính như sau:

1. Phần tiêu đề

Phần này bắt đầu bằng từ khoá PROGRAM tiếp theo là tên chương trình, sau tên chương tình là dấu ";". Sau tên chương trình ta có thể đưa vào các dòng chú thích hoặc lời giải thích, những dòng này không đặt cùng dòng với tên chương trình.

2. Phần khai báo

Có tất cả 7 tham số phải khai báo, tuy nhiên tuỳ từng chương trình chúng ta dùng loại nào thì khai báo loại đó, thứ tự các khai báo là:

- USES (CRT, GRAPH..... khai báo sử dụng các đơn vị chương trình)
- LABLE (khai báo nhãn)
- CONST (khai báo hằng)
- TYPE (Mô tả kiểu dữ liệu mới)
- VAR (khai báo biến)
- PROCEDURE (khai báo các thủ tục)
- FUNCTION (khai báo các hàm)

3. Phần thân chương trình

Phần thân chương trình phải đặt giữa hai từ khoá BEGIN và END., sau từ khoá END phải có dấu chấm để báo kết thúc chương trình. Nếu giữa chương trình có các đoạn chương trình có sử dụng BEGIN và END thì sau END sẽ là dấu ";".

Phần bắt buộc phải có trong một chương trình Pascal là phần thân chương trình bắt đầu bằng từ khoá BEGIN và kết thúc bằng từ khoá END. Những phần trên khi cần thì đưa vào không cần có thể bỏ kể cả dòng Program ... Như vậy một chương trình Pascal tối thiểu sẽ gồm ba dòng, hai dòng chứa các từ khoá Begin và End. Dòng còn lại chứa ít nhất một lệnh của Pascal .

Sơ đồ cấu trúc tổng thể của một chương trình Pascal như sau:

```
PROGRAM  Tên chương trình;

USES          Tên Unit ;
LABLE         (khai báo nhãn) ;
CONST          (khai báo hằng);
TYPE          (Mô tả kiểu dữ liệu mới);
VAR           (khai báo biến);

PROCEDURE   Tên thủ tục;
   Begin
     Thân chương trình con
   End;

FUNCTION   Tên hàm;
   Begin
     Thân chương trình con
   End;

BEGIN
 Thân chương trình chính;
END.
```

Khi viết chương trình chúng ta nên biết ngay một số tổ hợp phím đã được định nghĩa trong Pascal

Ctrl-K-B: đánh dấu đầu khối dòng

Ctrl-K-K: đánh dấu cuối khối dòng

Ctrl-Y: xoá một dòng ký tự tại vị trí con trỏ

Ctrl-Q-Y: xoá từ vị trí con trỏ đến cuối dòng

Ctrl-K-Y: xoá một khối đã đánh dấu

Ctrl-K-C: chép một khối đã đánh dấu đến vị trí mới của con trỏ

Ctrl-K-V: chuyển khối tới vị trí mới

Ctrl-K-W: ghi một khối vào một tệp mới

Ctrl-K-R: đọc một tệp khác từ đĩa vào văn bản hiện thời

Ctrl-Q-A: tìm kiếm và thay thế, các mục của tìm kiếm gồm:

U: tìm không phân biệt chữ hoa và chữ thường

B: tìm từ vị trí con trỏ về đầu tệp

G: tìm từ đầu tệp và thay thế

N: tìm và thay thế mà không cần hỏi lại

W: tìm một từ độc lập

Khi xây dựng một chương trình lớn, có những phần đã hoàn chỉnh và có thể sử dụng chung cho một số chương trình khác, người ta thường ghi chúng riêng thành các phần gọi là đơn vị chương trình (UNIT). Các đơn vị chương trình này có thể để riêng thành một File hoặc ghép chung thành một File lớn. Trong Turbo Pascal 7.0 có một số Unit như vậy ví dụ:

* GRAPH.TPU đây là một Unit về đồ hoạ, nó chứa rất nhiều chương trình con phục vụ việc vẽ và lựa chọn màu đồ hoạ.

* TURBO.TPL đây là tệp thư viện chương trình (TPL: Turbo Pascal Library), trong tệp này có thể gọi ra các Unit như:

- CRT: Unit quản lý màn hình, bàn phím

- PRINTER: Unit chứa các điều khiển việc in ấn

- TURBO3: Unit thiết lập sự tương thích giữa Turbo Pascal 3.0 với các Version cao hơn.

Muốn sử dụng một đơn vị chương trình nào ta phải đưa lời gọi ở đầu chương trình như sau: USES tên đơn vị chương trình; Để hiểu rõ thêm về các Unit xin mời tìm đọc cuốn "Lập trình nâng cao" của cùng tác giả.

CÂU HỎI ÔN TẬP CHƯƠNG 1

1. Nêu ý nghĩa của dấu cách, dấu gạch nối, dấu chấm phảy
2. Tên dùng để làm gì? Quy định của Pascal về cách đặt tên
3. Cho ví dụ về một số ký tự điều khiển của Pascal
4. Viết cấu trúc tổng thể của chương trình Pascal
5. Viết cấu trúc đơn giản nhất của chương trình Pascal
6. Cho biết ý nghĩa của từ khoá và tên chuẩn, nêu sự khác nhau giữa chúng
7. SQRT(n) là hàm khai căn bậc hai của số n đã được định nghĩa trong Pascal, chúng ta có thể định nghĩa lại để SQRT(n) cho ta Logarit cơ số 10 của n không?

Chương 2

KIỂU DỮ LIỆU, HẰNG, BIẾN, BIỂU THỨC, CÂU LỆNH

I. KIỂU DỮ LIỆU

Dữ liệu được hiểu là tất cả những gì mà con người đưa cho máy tính xử lý. Dữ liệu có thể là các con số, chữ viết, mệnh đề logic... Mỗi một phần mềm chỉ có thể xử lý được một vài loại dữ liệu cụ thể chứ không phải tất cả các loại. Tuy nhiên, máy tính như đã nói ở phần đầu chỉ hiểu được thông tin dưới dạng mã nhị phân, việc chuyển đổi thông tin sang hệ đếm cơ số 2 là công việc của các nhà thiết kế phần mềm, còn chúng ta chỉ cần quan tâm đến việc khai báo và sử dụng các dữ liệu cho đúng quy cách.

Dữ liệu trong ngôn ngữ Pascal được phân chia thành ba nhóm cơ bản sau đây (hình 2.1)

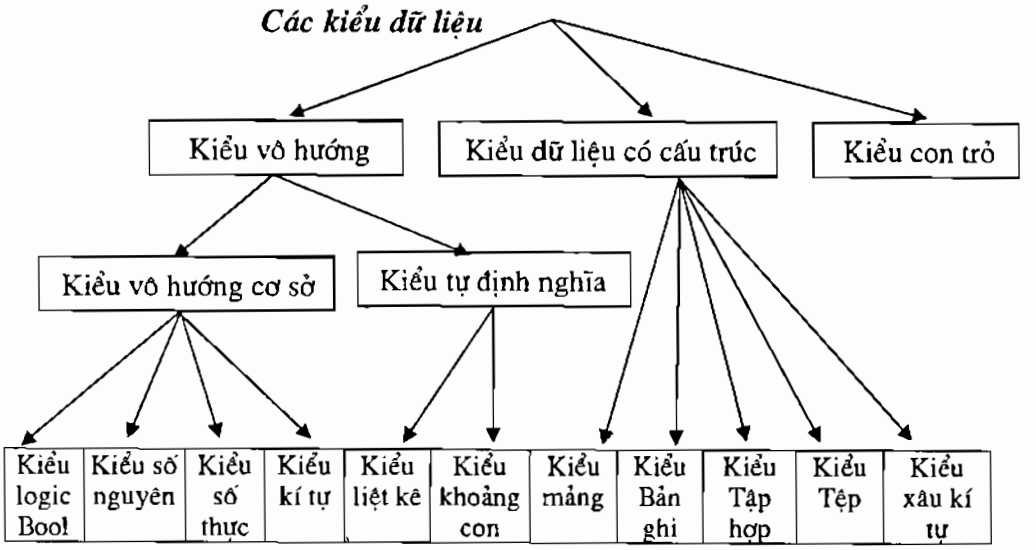
1. Kiểu vô hướng (Scalar type hay Simple type)

2. Kiểu dữ liệu có cấu trúc (Structured type)

3. Kiểu con trỏ (Pointer type)

Mỗi kiểu lại bao gồm một số kiểu cơ sở mà ta sẽ xem xét cụ thể trong các chương sau. Trong chương này ta sẽ khảo sát kiểu thứ nhất tức là kiểu vô hướng cơ sở.

Kiểu dữ liệu vô hướng là kiểu dữ liệu gồm một tập các giá trị được sắp xếp theo một thứ tự nào đó. Kiểu vô hướng bao gồm kiểu vô hướng chuẩn (Standard types) là kiểu đã được Pascal định nghĩa và kiểu do người sử dụng định nghĩa. Kiểu vô hướng chuẩn bao gồm bốn loại sau đây:



Hình 2.1

1. Kiểu lôgic Bun (Boolean)

Kiểu dữ liệu lôgic Bun là kiểu dữ liệu chỉ nhận một trong hai giá trị TRUE (đúng) hay FALSE (sai). Đây không phải là dữ liệu kiểu số nên các toán tử +, -,*,/.... không áp dụng được.

Các toán tử sử dụng với kiểu Bun bao gồm:

* AND: phép "và" logic

* OR: phép "hoặc" logic

* NOT: phép "đảo" hay "phủ định" logic

* XOR: phép "hoặc triệt tiêu"

- Phép AND cho kết quả TRUE khi và chỉ khi cả hai toán hạng đều là TRUE.

- Phép OR cho kết quả FALSE khi và chỉ khi cả hai toán hạng đều là FALSE.

- Phép XOR cho kết quả là TRUE khi hai toán hạng khác nhau, nếu hai toán hạng giống nhau thì kết quả luôn là FALSE.

Quy tắc các phép toán với các toán tử trên được ghi trong bảng 1 gọi là bảng chân lý của các toán tử (Truth Table).

Bảng 2.1

x	y	x and y	x or y	x xor y
false	false	false	false	false
false	true	false	true	true
true	false	false	true	true
true	true	true	true	false

x	not x
false	true
true	false

* Pascal quy định thứ tự của các ký tự và toán tử như sau:

- ❖ 1<2<3<4.....
- ❖ A<B<C<D<.....<Z
- ❖ a<b<c<d<.....<z
- ❖ FALSE<TRUE

Khi sử dụng các toán tử so sánh chúng ta luôn được kết quả là dữ liệu kiểu Boolean. Pascal có các toán tử so sánh sau đây:

<> : khác nhau , = : bằng nhau,

>= : lớn hơn hoặc bằng, <= : nhỏ hơn hoặc bằng,

> : lớn hơn , < : nhỏ hơn

Các toán hạng khi đưa vào biểu thức so sánh phải có cùng kiểu dữ liệu (trừ hai kiểu số nguyên và số thực : Integer - Real).

Ví dụ 2.1: các biểu thức so sánh sau:

12 < 30 cho kết quả TRUE

2*8 > 10 cho kết quả FALSE

FALSE < TRUE cho kết quả TRUE

'A' < 'B' cho kết quả TRUE

'BHF' >'BHFM' cho kết quả FALSE

2. Kiểu số nguyên (Integer)

Kiểu số nguyên sử dụng trong Pascal bao gồm 5 loại với các từ khoá khai báo khác nhau:

2.1 Kiểu BYTE

Dữ liệu kiểu BYTE được lưu trữ trong bộ nhớ bằng 1 byte, số nhị phân lớn nhất có thể xử lý là 1111 1111 tức là số 255 trong hệ thập phân, như vậy với những biến kiểu Byte chúng ta chỉ có thể nhập vào các giá trị từ 0 đến 255 .

2.2 Kiểu WORD

Kiểu WORD sử dụng 2 BYTES để lưu trữ dữ liệu và không xét dấu do đó các số có thể xử lý là từ 0 đến 65535

2.3 Kiểu INTEGER

Kiểu INTEGER là kiểu số nguyên có dấu sử dụng 2 byte, các số có thể xử lý là từ -32768 đến +32767 .

2.4 Kiểu SHORTINT

Dữ liệu kiểu SHORTINT được lưu trữ trong 1 byte nhưng là số có dấu do đó chỉ có thể xử lý các số trong phạm vi từ -128 đến +127

2.5 Kiểu LONGINT

Số nguyên kiểu LONGINT sử dụng 4 byte bộ nhớ trong đó bit bên trái nhất là bit dấu, phạm vi biểu diễn của LONGINT sẽ là từ -2147483648 đến 2147483647. Các toán tử thông dụng tác động lên dữ liệu kiểu số nguyên là:

Cộng " + "; trừ "-"; nhân "*"; chia "/" ; DIV; MOD

Chú ý: *Khi thực hiện các phép tính với những biến đã khai báo kiểu, cần lưu ý các kết quả trung gian để tránh các lỗi có thể xảy ra, ví dụ với khai báo biến x kiểu INTEGER, chúng ta thực hiện phép tính;*

x = 15000 + 20000 -12000 = 23000

Phép tính trên đây được thực hiện lần lượt từ trái qua phải do đó khi cộng 15000 với 20000 cho kết quả là 35000 vượt quá giá trị nguyên lớn nhất của kiểu Integer (32767) do đó máy sẽ báo lỗi.

* Hai phép tính đặc biệt dành riêng cho số nguyên là phép chia lấy phần nguyên DIV và phép chia lấy số dư MOD :

Ví dụ 2.2

17 DIV 7 = 2 (dư 3 không tính)

17 MOD 7 = 3 (phần nguyên = 2 không tính)

Số nguyên còn có thể làm đối số cho các hàm được định nghĩa trong Pascal ví dụ SIN(n), SQR(n) (hàm luỹ thừa 2), SQRT(n) (hàm khai căn bậc hai).... tuy nhiên khi khai căn bậc chẵn thì đối số phải là số dương.

Các số nguyên biểu diễn trong hệ nhị phân bởi các số 0 và 1, nếu ta xem số 0 tương ứng với giá trị FALSE và 1 tương ứng với giá trị TRUE thì có thể áp dụng các toán tử đã nêu trong kiểu Boolean cho kiểu số nguyên. Khi đó các toán tử AND, OR, XOR, NOT sẽ được gọi là toán tử số học. Quy tắc ứng dụng các toán tử này cũng giống như đã nêu trong bảng 1, nghĩa là:

0 AND 0 = 0 ; 1 AND 0 = 0; 0 AND 1 = 0 ; 1 AND 1 = 1

NOT(0) = 1 ; NOT(1) = 0

Với các số nguyên lớn 8 bit, 16 bit.. phép AND sẽ thực hiện qua từng bit như trong ví dụ sau:

Ví dụ 2.3:

3 = 0000 0011, 5 = 0000 0101

0000 0011 and 0000 0101 = 0000 0001

Tức là : 3 AND 5 = 1

Chú ý:

Khi viết các số nguyên âm trong hệ đếm cơ số mười thì dấu trừ phải viết liền ngay với các chữ số chứ không được để bất kỳ một khoảng cách nào.

3. Kiểu số thực (Real)

Kiểu số thực là tập hợp các số thực có thể biểu diễn được trong máy tính và được định nghĩa với từ khoá REAL. Dữ liệu kiểu Real sử dụng 6 bytes bộ nhớ. Pascal quy định giữa phần nguyên và phần thập phân là dấu "." . Số thực có thể viết dưới hai dạng:

3.1. Dạng thập phân bình thường

Đó là dạng viết quen thuộc mà chúng ta đã biết

Ví dụ: 3.1416, -1234.5678 ...

3.2 Dạng viết theo số mũ

** Trường hợp dấu phảy tĩnh*

Xét chữ số 12345.6789, chữ số này có thể viết dưới dạng mũ như sau:

12345.6789 = 1.23456789E+04

Trong đó phần số 1.23456789 với vị trí dấu chấm cố định sau số 1 được gọi là phần định trị,còn E+04 là luỹ thừa 4 của 10 được gọi là phần mũ.

** Trường hợp dấu phảy động*

Dấu phảy dùng để ngăn cách phần nguyên và phần thập phân, trong Pascal người ta đã dùng dấu phảy để ngăn cách các tên biến, tên hằng do đó trong số thực người ta

dùng dấu chấm thay dấu phảy. Do tính lịch sử nên ở đây chúng ta vẫn dùng từ "dấu phảy động" còn đúng ra thì phải nói là "dấu chấm động".

Ta có thể biểu diễn số 12345.6789 dưới các dạng khác như sau:

a. 12345.6789

b. 1234.56789E+01

c. 123.456789E+02

d. 12.3456789E+03

.....

Vị trí các chữ số tính từ phải qua trái được quy định là:

Số đầu tiên bên phải ở vị trí 0, số tiếp theo ở vị trí 1... Như vậy dấu chấm thập phân trong trường hợp a ở vị trí 4, trường hợp b ở vị trí 5,... Nói cách khác vị trí của dấu chấm đã thay đổi bên trong con số và ta nói đó là dấu chấm động.

Chú ý:

1. Khi đã định nghĩa một số là Real thì phải viết đầy đủ cả phần nguyên và phần lẻ Ví dụ: 0.345 hoặc 345.0 chứ không được viết .345 hoặc 345.

2. Các phép tính cộng, trừ, nhân, chia giữa một số nguyên và một số thực sẽ cho kết quả là một số thực.

3. Bình thường Pascal chỉ làm việc với kiểu Real, khi muốn nâng cao độ chính xác của phép tính Pascal cho phép sử dụng một số kiểu dữ liệu khác như trong bảng 2 dưới đây:

Bảng 2.2

Kiểu	Phạm vi biểu diễn	Chữ số có nghĩa	Số byte
Single	1.5*E-45 - 3.4E+38	7-8	4
Comp	-9.2E+18 - 9.2E+18	19-20	8
Double	5.0E-324 - 1.7E+308	15-16	8
Extended	3.4E-4932 - 1.1E+4932	19-20	10

Khi sử dụng một trong 4 kiểu trong bảng thì ở đầu chương trình cần đưa vào dẫn hướng biên dịch:

{$N+}

Nếu không có dẫn hướng này chương trình sẽ báo lỗi số 116:

Error 116: Must be in 8087 mode to compile this

Ví dụ 2.4:

```
uses crt;
var   x: real;
begin
clrscr;
```

```
x:=1000000/3;  writeln(x);
readln;
end.
```

Kết quả nhận được là: 3.3333333333E+05

Khi sử dụng dữ liệu kiểu dữ liệu Double với dẫn hướng dịch như trong ví dụ dưới đây thì kết quả nhận được sẽ là: 3.33333333333333E+0005

Ví dụ 2.5:

```
{$N+}
uses crt;
var
x:double;
begin
clrscr;
x:=1000000/3;
writeln(x);
readln;
end.
```

Việc tính toán với dữ liệu kiểu số bao gồm một số toán tử số học thông dụng đã giới thiệu, tuy nhiên để giúp người lập chương trình giảm bớt những công việc tính toán không cần thiết Pascal đã thiết kế sẵn một số hàm số học thông dụng (bảng 2.3) và người lập trình có thể sử dụng bất kỳ lúc nào.

Bảng 2.3

Tên hàm	Ý nghĩa
ABS(X)	|X| (giá trị tuyệt đối của X)
SQR(X)	X^2 (X bình phương)
SQRT(X)	$\sqrt{X}$ (căn bậc 2 của X)
LN(X)	lnX (logarit Nepe của X)
EXP(X)	e^X
SIN(X)	Sin(x)
COS(X)	Cos(x)
ARCTAN(X)	Arctang(x)
SUCC(n)	n+1 (số tiếp theo của n) , n nguyên
PRED(n)	n-1 (số kề trước của n) , n nguyên
TRUNC(X)	Bỏ phần lẻ lấy phần nguyên của X
ROUND(X)	Làm tròn phần lẻ của X đến giá trị nguyên gần nhất

4. Kiểu ký tự (character)

Ký tự được hiểu là tất cả các chữ cái, chữ số, các dấu hiệu phân cách mà chúng ta sử dụng để viết văn bản. Trong Pascal kiểu dữ liệu ký tự được định nghĩa bởi từ khoá CHAR. Mỗi hằng ký tự khi khai báo phải viết giữa hai dấu nháy ví dụ 'A', '3', ... Hiện nay các phần mềm thông dụng trên thế giới đều sử dụng bảng mã ký tự của ASCII (Americain Standard Code for Information Interchange). Do vậy việc khai báo biến chỉ có thể chọn các ký tự của ASCII chứ không thể chọn các ký tự tiếng Việt.

Các ký tự trong bảng mã ASCII được đánh số thứ tự từ số 0, Pascal đã thiết kế hai hàm chuẩn ORD và CHR để xác định mối tương quan giữa số thứ tự của một ký tự và dạng biểu diễn ký tự trên màn hình.

Hàm ORD('A')=65, ORD('Z')=90

Hàm CHR(65)='A' , ..., CHR(90)='Z'

Để xác định ký tự trước hoặc sau ký tự hiện thời trong bảng mã ASCII có thể dùng hai hàm đã có sẵn là Pred (trước) và Succ (sau).

Ví dụ:

Pred('C') cho kết quả là 'B'

Còn Succ('C') cho kết quả là 'D'

Chú ý:

1. Một biến khai báo dạng CHAR khi gán giá trị cho biến chỉ có thể gán từng ký tự chứ không thể gán cả một từ, cách gán như sau là sai

```
Var
làm:char;
begin
lam:='yes';
.....
```

2. Một số ký tự trong bảng mã là các ký tự điều khiển do đó chúng ta không thể cho hiện các ký tự này trên màn hình, ví dụ:

Ký tự thứ 10 là ký tự điều khiển xuống dòng (Line Feed)

Ký tự thứ 13 là ký tự nhảy về đầu dòng (Carriage Return)

Ví dụ 2.6:

Chương trình sau sẽ cho hiện lên màn hình các ký tự của bảng mã ASCII. Cách thức viết chương trình và ý nghĩa của các lệnh sẽ được giới thiệu trong các phần tiếp theo của giáo trình.

```
Program ma_ascii;
Uses crt;
 Var i:byte;
 BEGIN
 Clrscr;
 gotoxy(30,1);
 Writeln('CAC KY TU BANG MA ASCII');
```

```
writeln;
 for i:=0 to 255 do write('[',i,']',chr(i),' ');
ReadIn;
END.
```

Chạy chương trình trên chúng ta sẽ không thấy được trên màn hình các ký tự từ ký tự số 10 đến ký tự số 13.

5. Kiểu liệt kê

Là kiểu dữ liệu vô hướng mới do người lập trình tự đặt tên. Các thành phần của kiểu dữ liệu này được lấy ra từ các kiểu vô hướng chuẩn bằng cách liệt kê các giá trị của kiểu vô hướng. Danh sách các giá trị này được đặt trong ngoặc đơn và được mô tả bằng một tên kiểu trong phần mô tả (Phần TYPE).

Ví dụ 2.7:

```
TYPE
Boolean=(False, True);
Color=(Red, Blue, Green, While, Black);
```

Một biến vô hướng có thể định nghĩa thông qua các kiểu đã được mô tả trong phần Type như sau:

```
VAR
Ketqua: Boolean;      Mau1, Mau2: Color;
```

Pascal còn cho phép khai báo trực tiếp biến kèm theo mô tả kiểu dữ liệu:

```
VAR
 Gioitinh: (Nam, nu);
 Ngay: (Chunhat, Hai, Ba, Tu, Nam, Sau, Bay);
```

Sau khi đã khai báo biến vô hướng, có thể dùng các phép gán hoặc nhập dữ liệu từ bàn phím, ví dụ:

```
Ketqua:= True;
Mau1:=Blue;
Gioitinh:=Nam;
Ngay:=Chunhat;
```

6. Kiểu khoảng con

Là kiểu vô hướng được ứng dụng khi một biến chỉ được nhận các giá trị trong một khoảng (xác định bởi cận trên và cận dưới). Kiểu dữ liệu này tiết kiệm được bộ nhớ và thường được sử dụng để kiểm tra xem các giá trị nhập cho biến có vượt khỏi phạm vi quy định hay không.

Ví dụ 2.8:

```
TYPE
Ngay=(Chunhat, Hai, Ba, Tu, Nam, Sau, Bay);
      Chuhoa='A'..'Z';
```

```
VAR
    Chu: Chuhoa;
    Ngay_lam_viec: Hai..Bay;
```

Trong ví dụ 2.8 chúng ta đã định nghĩa kiểu dữ liệu liệt kê **Ngay** và kiểu dữ liệu khoảng con **Chuhoa.** Sau đó đã khai báo hai biến, biến "Chu" thuộc kiểu dữ liệu Chuhoa và biến Ngay_lam_viec là kiểu khoảng con của "Ngay".

II. HẰNG, BIẾN, KIỂU DỮ LIỆU MỚI

Hằng, biến và kiểu dữ liệu do người lập trình định nghĩa bao giờ cũng phải khai báo ở đầu chương trình. Mỗi loại có một từ khoá riêng đã được chuẩn hoá trong Pascal chúng ta không được viết sai hoặc sử dụng các từ khoá này vào mục đích khác.

1. Khai báo hằng

Hằng số là các đại lượng mà giá trị của nó không thay đổi trong suốt quá trình xử lý. Để khai báo hằng ta dùng từ khoá CONST.

Ví dụ 2.9:

```
CONST
pi = 3.1416; (hằng kiểu Real);
lam = 'c'; (hằng kiểu Char)
suthat = TRUE; (hằng kiểu Boolean)
```

2. Khai báo biến

Biến (Variable) là những đại lượng mà giá trị của nó có thể thay đổi trong quá trình xử lý. Giá trị của mỗi biến được lưu trữ trong một ô nhớ của bộ nhớ trong của máy. Để khai báo biến ta dùng từ khoá VAR. Nếu có nhiều biến cùng kiểu ta có thể viết trên cùng một dòng, mỗi biến cách nhau một dấu phảy, giữa phần tên biến và phần kiểu dữ liệu là dấu hai chấm.

Ví dụ 2.10:

```
VAR
x1,x2,x3: Real; (các biến x1, x2, x3 là biến kiểu số thực)
a,b,c: Integer (a, b, c là biến nguyên)
```

Pascal cho phép người sử dụng vừa khai báo biến vừa gán giá trị ban đầu cho biến, việc này phải sử dụng từ khoá Const.

Ví dụ 2.11:

```
CONST
a: real= 123.45;   lam : char='Y';
```

3. Định nghĩa kiểu dữ liệu mới

Ngoài những kiểu dữ liệu Pascal đã định nghĩa và chúng ta đã khảo sát trong chương 2, người lập trình có thể tự định nghĩa một số kiểu dữ liệu khác tuỳ theo yêu cầu của bài toán. Kiểu dữ liệu mới được khai báo bằng từ khoá TYPE ở phần đầu của chương trình.

Ví dụ 2.12:

```
TYPE
mau = (xanh, do, tim, vang, den, xam);
VAR
mauchu: mau;
```

Trong ví dụ trên kiểu dữ liệu 'mau' là một kiểu dữ liệu đếm được gồm 6 tham số tên màu, biến 'mauchu' là biến kiểu 'mau' nghĩa là nó có thể nhận một trong các giá trị xanh, do, tim,...

4. Biểu thức

Biểu thức được hiểu là một công thức tính toán theo một quy tắc nào đó. Biểu thức bao gồm các toán tử +, -, *,/, =, >, <.... và các toán hạng, toán hạng ở đây có thể là các hằng, biến, hàm đã được định nghĩa trong Pascal .

Biểu thức được chia thành hai loại:

* *Biểu thức số học*: là biểu thức cho ta giá trị bằng số

Ví dụ 7: (3 + 5)/2 kết quả bằng 4

* *Biểu thức Logic:* là biểu thức cho ta kết quả logic tức là một trong hai giá trị FALSE hoặc TRUE.

Ví dụ: 3 > 5 cho kết quả FALSE

Chú ý:

Trong các biểu thức Logic có sử dụng toán tử so sánh <, >, >=, <=, =, <> thì các toán hạng phải tương thích với nhau về kiểu.

Ví dụ 2.13:

A < B cho giá trị TRUE

A < 5 là biểu thức không hợp lệ vì A là kiểu Char còn 5 là kiểu Integer.

Thứ tự ưu tiên các phép toán trong Pascal được quy định như sau:

Toán tử	Mức ưu tiên
(..)	1
NOT, -	2
*, /, DIV, MOD, AND	3
+, -, OR, XOR	4
=, <>, <=, >=, >, <, IN	Toán tử quan hệ cùng mức ưu tiên

Ví dụ 2.14: Tính giá trị của biểu thức Logic sau:

Not (30*2>50) Or ('B'<'A') And (2>5 div 2)

Thực hiện các phép toán trong ngoặc trước, ta có

Not (True) Or (False) And (False)

Thực hiện And trước or:

False Or False

Kết quả cuối cùng là:

False

5. Câu lệnh

Các câu lệnh được viết ở phần thân của chương trình, câu lệnh bao gồm các lệnh đơn hoặc lệnh có cấu trúc. Câu lệnh nhằm thực hiện một công việc nào đó mà người lập trình đặt ra cho máy xử lý. Việc bố trí mỗi câu lệnh trên một dòng hay nhiều câu lệnh trên một dòng là tuỳ cách trang trí của người lập trình miễn là các câu lệnh phải đặt cách nhau bởi dấu phân cách ";".

Ví dụ 2.15:

```
CONST
a = 3.14;
VAR
X, DIENTICH: Real;
Y: Boolean;
BANKINH: Integer;
BEGIN
(* Tính diện tích hình tròn *)
BANKINH:=5;
DIENTICH:=BANKINH*BANKINH*a;
Writeln(' Dien tich hinh tron = ',dientich:5:2);
END.
```

Chú ý:

Các từ khoá trong Pascal có thể viết chữ thường, chữ hoa hoặc xen kẽ chữ thường và chữ hoa máy đều chấp nhận.

Ví dụ: CONST hay COnst hay coNSt đều như nhau

6. Phép gán giá trị

Để gán giá trị cho hằng hoặc biến ta dùng cụm ký tự ":=", cần lưu ý phân biệt phép gán với phần khai báo hằng sau từ khoá CONST. Trong phần khai báo ta chỉ dùng ký tự "=".

Phép gán " := " được hiểu là lấy giá trị viết **bên phải** dấu := gán cho biến viết ở **bên trái.**

Giá trị đem gán có thể là hoàn toàn mới, cũng có thể là giá trị đã có sẵn trong biến hay trong các phần tử mảng.

Ví dụ:

```
X := 5;          (* gán một giá trị mới cho biến X *)
X := X+3;        (* lấy giá trị đã có trong X cộng với 3 rồi gán vào X, sau phép gán
này biến X sẽ mang giá trị là 8 *)
```

Khi gán giá trị phải bảo đảm tính tương thích giữa kiểu dữ liệu và giá trị gán, ví dụ nếu ta khai báo X là biến kiểu INTEGER thì không thể gán cho nó X:=6.5.

Trong các phép gán có một ngoại lệ đó là biến thực (REAL) có thể nhận các giá trị nguyên.

CÂU HỎI ÔN TẬP CHƯƠNG 2

1. Tên các kiểu dữ liệu trong chương này có thể thay đổi với ý nghĩa khác đi hay không?
2. Tại sao bảng mã ASCII lại chỉ có 256 ký tự.
3. Sự giống và khác nhau giữa hai kiểu Integer và Word.
4. Một lớp học có 50 học sinh, nên dùng kiểu dữ liệu gì để đánh số thứ tự?
5. Để gán các giá trị: 8, 2.5, 'C' lần lượt cho các biến x, y, z thì x, y, z phải thuộc kiểu dữ liệu gì? Viết lệnh gán.
6. T là một biến kiểu Longint, x, y, z lấy trong câu 4, các câu lệnh sau đây cho kết quả như thế nào:

   T := x*1000000;

   T := y*1000000;

   Y:= x*x-y;

   Z := x-y;
7. Có thể gán các ký tự ă, â, ô, ê, ư cho một biến kiểu Char hay không?
8. Số nguyên nhỏ nhất và lớn nhất mà Pascal có thể xử lý là bao nhiêu?
9. Có những biểu thức loại gì trong Pascal ?
10. Có thể gán giá trị mới cho một hằng đã khai báo hay không?
11. Biểu thức có phải là một câu lệnh không?

Chương 3

THỦ TỤC NHẬP, XUẤT DỮ LIỆU

Nhập dữ liệu có nghĩa là gán cho hằng, biến, các phần tử mảng... những giá trị phù hợp nào đó. Xuất dữ liệu nghĩa là viết ra màn hình hoặc máy in những số liệu đang lưu trữ trong các biến. Việc đưa dữ liệu ra máy in về cơ bản không khác nhiều so với đưa ra màn hình do vậy chúng ta tạm thời xét trường hợp viết ra màn hình. Màn hình máy vi tính hiện nay ở chế độ ngầm định được chia thành 25 dòng và 80 cột. Trong chế độ đồ hoạ màn hình được chia thành một ma trận điểm, với màn hình VGA số điểm theo chiều ngang thường là 640 và theo chiều đứng là 480. Với những màn hình độ phân giải cao tỷ lệ này có thể là 800/600 hoặc 1024/768.

I. NHẬP DỮ LIỆU

Các đại lượng thường phải nhập dữ liệu trong Pascal bao gồm:

Hằng, Biến, phần tử mảng, trường trong bản ghi...

Có hai cách nhập dữ liệu là nhập trực tiếp bằng lệnh gán := và nhập bằng cách gõ số liệu từ bàn phím. Phép gán đã được giới thiệu trong chương 2 ở đây chúng ta sẽ chỉ đề cập đến cách thứ hai.

Để gán giá trị cho một biến nào đó có thể sử dụng một trong hai thủ tục chuẩn của Turbo Pascal là READ(..) và READLN(...).

1. Thủ tục READ(...) và READLN(...)

Thủ tục Read được sử dụng để đọc dữ liệu từ tệp ra biến hoặc nhập dữ liệu từ bàn phím vào cho biến nếu tên biến đã được chỉ rõ.

Thủ tục Read(tên biến) đòi hỏi có các tham số là biến đã được khai báo. Khi gặp thủ tục này Pascal sẽ tạm dừng chương trình và con trỏ sẽ nhấp nháy chờ trên màn hình, dữ liệu gõ từ bàn phím sẽ hiện trên màn hình và sẽ được nhập vào biến khi chúng ta bấm phím Enter. Khi việc nhập dữ liệu đã hoàn thành chương trình sẽ tiếp tục thực hiện các lệnh tiếp theo.

Có thể sử dụng thủ tục READ(...) để nhập dữ liệu cho nhiều biến cùng một lúc, tên các biến phải đặt cách nhau bởi dấu phảy.

Ví dụ 3.1:

Để nhập các giá trị 1, 4.5, 2 cho các hệ số a, b, c khi giải phương trình bậc hai chúng ta viết lệnh:

```
Read(a,b,c);
```

Trên màn hình các số 1, 4.5, 2 phải viết cách nhau ít nhất là một khoảng cách và tuyệt đối không dùng các dấu khác để ngăn cách.

Với các con số đã gõ Pascal hiểu là mang số 1 gán vào biến a, số 4.5 gán vào b và số 2 gán vào c.

Nếu chúng ta quên không gõ khoảng cách giữa các số liệu, ví dụ giữa số 4.5 và số 2:

1 4.52

thì khi bấm Enter con trỏ sẽ chuyển xuống dòng dưới nhưng vẫn nhấp nháy chờ trên màn hình bởi lẽ Pascal cho rằng số 1 nhập vào cho a, số 4.52 nhập vào cho b, còn biến c chưa có dữ liệu. Gặp trường hợp này chúng ta đành phải gõ một giá trị nào đó cho c để kết thúc nhập dữ liệu và chạy lại chương trình sau.

Vẫn trường hợp trên sau khi gõ 1 4.52 nếu vô ý chúng ta gõ nhiều lần phím Enter thì con trỏ sẽ chuyển dần xuống đáy màn hình và có thể màn hình trở nên tối đen chỉ còn lại con trỏ nhấp nháy. Gặp trường hợp này xin đừng vội tắt máy, hãy gõ một số hoặc một ký tự bất kỳ rồi bấm Enter, có thể Pascal sẽ thông báo lỗi, nhưng đổi lại chúng ta đã trở về được màn hình soạn thảo chương trình.

Có một vấn đề cần đặc biệt lưu ý là các biến trong lệnh Read có thể thuộc các kiểu dữ liệu khác nhau do đó khi gõ dữ liệu từ bàn phím cần gõ đúng kiểu dữ liệu.

```
VAR  x,y:integer;  x1,x2,: char;
Begin
Write('Nhap gia tri x,y,x1,x2');  Readln(x,y,x1,x2);
End.
```

Chạy chương trình trên màn hình sẽ xuất hiện câu thông báo:

Nhap gia tri x,y,x1,x2

Nhập từ bàn phím lần lượt các số liệu 123 4567 a b ↵

Sau phép đọc trên biến x có giá trị 123, biến y có giá trị 4567 còn biến x1 là a và x2 là b.

Trong chương trình thủ tục READLN sẽ đọc dữ liệu cho lần lượt các biến x, y, x1, x2, nếu chúng ta nhập số đầu tiên không phải là số nguyên ví dụ là 12.45 thì máy sẽ báo lỗi vì x là biến nguyên không thể nhận giá trị thực.

Riêng đối với dữ liệu kiểu chuỗi ký tự (String) khi nhập cho nhiều biến cùng một lúc thì số ký tự gõ từ bàn phím cho một biến nào đó phải bằng *chiều dài tối đa* đã khai báo, ký tự khoảng cách ở đây được Pascal coi là một ký tự thuộc chuỗi chứ không phải là dùng để phân biệt các số liệu cho biến. Để thấy rõ vấn đề này ta xét ví dụ sau (một số lệnh trong ví dụ sẽ giới thiệu trong các phần tiếp theo):

Ví dụ 3.2:

```
uses crt;
var a1,a2:string[3];
begin
clrscr;
readln(a1,a2);
textcolor(4);
writeln(a1);
```

```
textcolor(14);
writeln(a2);
readln;
end.
```

Chương trình trên khai báo hai chuỗi a1, a2 có độ dài tối đa là 3 ký tự. Lệnh clrscr sẽ xoá sạch màn hình đưa con trỏ về toạ độ 1,1. Lệnh Readln(a1,a2) dùng để nhập dữ liệu vào cho hai chuỗi. Lệnh Textcolor(..) định màu chữ sẽ viết trên màn hình, số 4 là màu đỏ, số 14 là màu vàng. Lệnh Writeln(..) sẽ viết dữ liệu lưu trữ trong các biến lên màn hình với màu quy định.

Chạy chương trình và nhập vào

ABCDE ↵

Kết quả trên màn hình ta nhận được một dòng là chữ ABC màu đỏ, còn dòng dưới là chữ DE màu vàng, như vậy là Pascal đã lấy 3 ký tự đầu ABC gán cho a1, còn lại 2 ký tự DE gán cho a2.

Chạy chương trình lần thứ hai và nhập vào

AB CDE ↵ (sau chữ B là khoảng cách)

Kết quả là dòng phía trên có 2 ký tự AB màu đỏ còn dòng dưới là 3 ký tự CDE màu vàng. Thực ra dòng trên vẫn là 3 ký tự nhưng ký tự thứ ba là khoảng cách nên ta không nhìn thấy được.

Chạy chương trình lần thứ ba và nhập vào

ABC DE ↵ (sau chữ C là hai khoảng cách)

Kết quả dòng trên là ABC màu đỏ còn dòng dưới chỉ thấy một chữ D màu vàng.

Tóm lại để khỏi bị nhầm lẫn khi nhập dữ liệu cần tuân thủ các quy tắc sau đây:

1. Thông báo tên các biến cần nhập dữ liệu trên màn hình
2. Dữ liệu nhập cho các biến cần đặt cách nhau ít nhất một khoảng cách
3. Nhập xong dữ liệu phải bấm phím Enter
4. Biến kiểu chuỗi khi nhập phải đặt trong cặp dấu nháy, ví dụ:

S:='Viet nam';

Không nên nhập nhiều biến kiểu chuỗi trong cùng một lệnh Read.

Thủ tục Read(...) không thể nhập dữ liệu cho Hằng hoặc Biểu thức.

Thủ tục Readln(...) cũng dùng để nhập dữ liệu như thủ tục Read(...) sự khác nhau giữa hai thủ tục này là: với thủ tục READ (của các Version nhỏ hơn 7.0) sau khi ấn Enter để kết thúc nhập dữ liệu con trỏ không chuyển xuống dòng dưới, còn với READLN con trỏ sẽ chuyển xuống dòng tiếp theo. Riêng Version 7.0 thì Read và Readln là như nhau.

Ngoài hai thủ tục nêu trên còn thủ tục READLN , thủ tục này không có tham số nghĩa là không nhập dữ liệu nào vào máy, khi đó máy sẽ chờ để ta gõ phím Enter. Thực chất đây là một lệnh dừng chương trình để ta kiểm tra hoặc quan sát màn hình, sau khi kiểm tra xong bấm Enter để tiếp tục chương trình.

2. Thủ tục Readkey và Keypessed

Trong qua trình nhập dữ liệu cũng như xây dựng chương trình người ta hay dùng hai thủ tục đặc biệt của Turbo Pascal là ReadKey (đọc một ký tự từ bàn phím) và KeyPressed (bấm một phím nào đó trên bàn phím).

Cú pháp của thủ tục Readkey như sau:

Tên biến := Readkey;

Nếu một biến ký tự được gán giá trị thông qua thủ tục Readkey thì khi gặp thủ tục này máy sẽ chờ ta gõ một ký tự từ bàn phím. Vì chỉ gõ một ký tự nên biến phải thuộc kiểu dữ liệu Char.

Trong các chương trình thủ tục Readkey thường được dùng để trả lời câu hỏi có 'Y' hay không 'N'. Căn cứ vào phím vừa gõ máy sẽ quyết định công việc tiếp theo.

Ví dụ 3.3:

```
Var
lam:char; x,y:real;
Begin
Repeat
Readln(x,y);
lam:=Readkey;
until lam in ['k', 'K'];
End.
```

Câu lệnh Repeat....Until ... trong ví dụ 3.3 là một lệnh lặp ta sẽ nghiên cứu ở phần sau. Biến "lam" là biến ký tự. Sau khi gán giá trị cho hai biến x, y gặp lệnh lam:= Readkey máy sẽ chờ, nếu ta gõ vào phím k thì công việc kết thúc (không phân biệt chữ k hoa hay chữ thường), còn nếu gõ vào một phím bất kỳ thì quá trình sẽ lặp lại nghĩa là lại nhập dữ liệu vào hai biến x,y.

Thủ tục KeyPressed (bấm một phím) thường được dùng làm điều kiện trong một câu lệnh. Gặp thủ tục này máy sẽ chờ cho đến khi một phím bất kỳ được bấm.

Hai thủ tục Readln và Keypressed đều được dùng để giữ màn hình lại cho ta quan sát, sự khác biệt giữa là ở chỗ khi xem xong để tiếp tục công việc với Readln ta cần bấm Enter còn với Keypressed ta bấm một phím bất kỳ.

Ví dụ 3.4:

```
Var
lam:char;
Begin
clsscr; (* xoá sạch màn hình *)
writeln('Ban hay bam phim Enter hoặc Esc ');
lam:=readkey; (* gán cho biến lam giá trị từ bàn phím *)
Case Ord(lam) of
13: Writeln('Ban vua bam phim Enter');
```

```
27: Writeln('Ban vua bam phim Esc');
end;
Repeat until Keypressed;
End.
```

Trong ví dụ trên biến "lam" được gán giá trị từ bàn phím thông qua thủ tục Readkey. Hàm Ord(lam) cho biết số thứ tự của phím vừa được bấm (xem phần giới thiệu hàm Ord ở chương 2). Cấu trúc Case ... of... là cấu trúc rẽ nhánh chúng ta sẽ xét sau. Phím Enter có số thứ tự là 13, phím Esc có số thứ tự là 27. Nếu ta bấm phím Enter thì máy sẽ viết ra câu thông báo "Ban vua bam phim Enter" còn nếu ta bấm phím Esc thì máy sẽ thông báo "Ban vua bam phim Esc". Dòng lệnh Repeat until Keypressed; sẽ giữ nguyên màn hình cho đến khi ta bấm một phím bất kỳ nào đó trên bàn phím.

II. VIẾT DỮ LIỆU RA MÀN HÌNH

1. Thủ tục Write(...) và Writeln(...)

Có ba thủ tục viết dữ liệu ra màn hình là WRITE(...), WRITELN(...) và WRITELN.

Hai thủ tục WRITE(...), WRITELN(...) đòi hỏi có tham số đi kèm. Tham số trong các lệnh trên có thể là tên các biến, tên hằng, biểu thức hoặc các phần tử của mảng, chuỗi. Nếu muốn viết nhiều đại lượng ra màn hình thì cần dùng dấu phảy để ngăn cách tên các biến hoặc các biểu thức

Với các hằng ký tự cần phải đặt chúng trong cặp dấu nháy đơn '...'.

Sự khác nhau giữa ba thủ tục trên là:

WRITE(...) sau khi viết xong giá trị cuối cùng con trỏ nằm nguyên tại vị trí kết thúc không chuyển xuống dòng dưới.

WRITELN(...) sẽ viết giá trị các biến trên cùng một dòng sau khi viết xong con trỏ sẽ được tự động chuyển xuống đầu dòng tiếp theo.

WRITELN không viết gì ra màn hình mà chỉ chuyển con trỏ xuống đầu dòng dưới.

Để viết dòng chữ "Chuc mung nam moi" màu đỏ trên nền vàng bắt đầu tại vị trí dòng 10 cột 30 ta phải viết một chương trình (ví dụ 3.5)

Ví dụ 3.5:

```
Program Vietchu;
uses crt;
Begin
gotoxy(30,10);
textcolor(red);
textbackground(yellow);
write('Chuc mung nam moi');
End.
```

2. Viết các ký tự ra màn hình

Các ký tự khi viết ra màn hình phải đặt trong dấu ngoặc đơn, nếu không có các lệnh gì khác thì ký tự được viết bắt đầu từ vị trí hiện tại của con trỏ và chiếm một khoảng rộng đúng bằng độ dài của chuỗi.

Nếu trước khi viết trong chương trình có lệnh CLRSCR (xoá sạch màn hình) thì các ký tự sẽ được viết ra tại toạ độ 1,1, còn nếu trước đó lại là lệnh WRITELN (xuống dòng) thì ký tự sẽ được viết ở đầu dòng mới.

Sử dụng cách viết có quy cách nghĩa là quy định độ dài chuỗi sẽ viết ra thì ký tự cuối cùng của chuỗi sẽ được căn theo lề phải trong phạm vi độ rộng quy định

Ví dụ 3.6:

```
Var
lam:char;
Begin
lam:='Y'
Writeln(lam);
Writeln(lam:7);
Writeln('ABCDE')
Writeln('ABCDE':8);
End.
```

Kết quả sẽ viết ra màn hình như sau:

```
Y
      Y
ABCDE
    ABCDE
Vị trí 1 2 3 4 5 6 7 8
```

Nếu trong lệnh Writeln('ABCDE':n) ta chọn n nhỏ hơn độ dài chuỗi (ở đây là 5) thì máy vẫn viết ra đầu đủ cả chuỗi.

Để viết một ký tự nào đó trong bảng mã ASCII ta cần biết số thứ tự của nó trong bảng. Một số ký tự đầu trong bảng mã là các ký tự điều khiển không in ra được. Ví dụ ký tự số 7 là tiếng chuông, nếu ta viết lệnh

Write(char(7)) thì máy sẽ phát một tiếng chuông.

Lệnh trên cũng có thể viết dưới dạng sau Write(#7)

3. Viết số ra màn hình

Cách viết các số nguyên và số thực là khác nhau.

* Trường hợp không định kiểu

Với các số nguyên máy sẽ viết ra đầy đủ số chữ số kể cả dấu bắt đầu từ vị trí hiện thời của con trỏ.

Với các số thực máy sẽ dành 18 vị trí để viết mỗi số theo quy định sau: 2 vị trí đầu để trống, tiếp đến một số phần nguyên, dấu chấm phân cách, 10 vị trí phần thập phân, chữ E biểu hiện phần mũ, dấu của mũ và hai vị trí cho số mũ.

Ví dụ 3.7:

```
Writeln(-123456);
Write(123.4567);
```

Kết quả:

```
-123456
 1.2345670000E+02
```

* Trường hợp có định kiểu

Qua ví dụ trên rõ ràng là cách viết số thực không định kiểu sẽ rườm rà khó đọc và khó trình bày trên màn hình cũng như in ra giấy, vì vậy cần phải quy định cách viết theo ý người lập trình.

Cách viết có định kiểu đòi hỏi chỉ cho máy biết số vị trí mà máy phải viết dữ liệu ra tính từ vị trí hiện thời của con trỏ. Riêng đối với số thực thì phải quy định rõ số vị trí cho toàn bộ số và số vị trí cho phần thập phân.

Ví dụ 3.8:

```
Write(' So thu tu : '); Writeln(381:14);
Write(' Ho va ten: Nguyen Van Tan ');
Write(' Luong    : '); Writeln(359000.123:12:4);
Write(' Phu cap  : '); Writeln(2400.65:12:4);
```

Kết quả viết ra màn hình như sau:

```
 So thu tu    :          381
 Ho va ten  : Nguyen Van Tan
 Luong        :  359000.1230
 Phu cap      :    2400.6500
```

Trong ví dụ trên ở dòng thứ hai tên người chiếm 14 ký tự tính từ dấu : , lệnh viết ở dòng thứ 3 và 4 ta quy định cho máy viết số thực với độ rộng tổng thể là 12 chữ số trong đó có 4 số lẻ , vì vậy các số bắt đầu viết tại vị trí dưới chữ T của từ Tan . Nếu muốn các chữ và số trên các dòng đều dóng thẳng lề phải từ vị trí chữ "n" trong từ "Tan" thì ta phải quy định độ rộng là 14.

Nếu trong lệnh viết số thực ta chỉ quy định đồ dài tổng thể mà không quy định số vị trí cho phần thập phân thì máy sẽ viết ra theo kiểu mũ trong đó 4 vị trí dành cho phần số mũ, phần số sẽ được làm tròn với độ rộng còn lại.

Ví dụ 3.9:

```
Write(123.456789:8);
```

Kết quả:

```
1.23E+02
```

Trong quá trình nhập dữ liệu, để tránh nhầm lẫn nên có những thông báo cần thiết về mỗi biến sẽ nhập, có thể dùng kết hợp lệnh viết Write và lệnh Read như trong chương trình sau đây:

Ví dụ 3.10:

```
Uses crt;
Var luong,phucap: Real;  stt:Integer;
Begin
Clrscr;
Write('Nhap so thu tu: '); Readln(Stt);
Write('Nhap luong : '); Readln(luong);
Write('Nhap phu cap : '); Readln(phucap);
End.
```

Thực hiện chương trình ta sẽ thấy xuất hiện đầu tiên là dòng chữ

Nhap so thu tu:

Như vậy ta sẽ biết ngay công việc bây giờ là nhập vào số thứ tự, giả sử gõ vào số 1 và bấm Enter ta sẽ thấy xuất hiện tiếp :

Nhap luong: (gõ vào 850000 và bấm ↵)

Nhap phu cap: (gõ vào 50000 và bấm ↵)

Mỗi câu thông báo xuất hiện trên màn hình bên cạnh sẽ có con trỏ nhấp nháy chờ để ta nhập dữ liệu, nhập xong con trỏ sẽ tự động chuyển xuống dòng dưới vì lệnh nhập dữ liệu là Readln.

4. Đưa dữ liệu ra máy in

Pascal quy định máy in là LST, việc quản lý máy in đã được chuẩn hoá và cài đặt trong Unit Printer.

Để đưa dữ liệu ra máy in trước hết phải gọi Unit Printer bởi lệnh:

USES PRINTER;

Tiếp đó phải bật máy in và lắp giấy in trước khi thực hiện các lệnh in.

Cú pháp của lệnh in là:

WRITE(LST, tên biến/giá trị);

Ví dụ 3.10 đã nhập dữ liệu vào cho các biến STT, LUONG, PHUCAP để in các dữ liệu này ra giấy cần bổ sung vào chương trình các lệnh in (xem ví dụ 3.11).

Ví dụ 3.11:

```
Uses crt;
Var luong,phucap:Real;  stt:Integer;
Begin
clrscr
Write('Nhap so thu tu: '); Readln(Stt);
Write('Nhap luong : '); Readln(luong);
```

```
Write('Nhập phu cap : '); Readln(phucap);
Clrscr;
Writeln(Lst, '  DU LIEU DA NHAP VAO CHO CAC BIEN ');
Writeln(lst);
Writeln(Lst, ' So thu tu : ',stt);
Writeln(Lst, ' Luong     : ',luong:10:2);
Writeln(Lst, ' Phu cap   : ',phucap:10:2);
Readln;
End.
```

Khi thực hiện chương trình trước tiên máy yêu cầu nhập dữ liệu, chúng ta sẽ gõ như trong ví dụ 3.10. Tiếp theo màn hình sẽ bị xoá sạch và hiện lên kết quả thực hiện bởi các lệnh Writeln(...), đồng thời máy in cũng hoạt động và in ra kết quả giống như trên màn hình.

```
So thu tu : 1
Luong     : 850000.00
Phu cap   :  50000.00
```

III. THỦ TỤC QUẢN LÝ MÀN HÌNH

Các thủ tục trình bày màn hình trong Turbo Pascal 7.0 được ghi thành một đơn vị chương trình (Unit) có tên là CRT, Unit này đặt trong tệp TURBO.TPL, vì vậy muốn trình bày màn hình ở đầu chương trình phải khai báo: USES CRT;

Một số thủ tục thông thường hay sử dụng:

GOTOXY(x,y): đưa con trỏ màn hình đến vị trí có toạ độ (x,y) trong đó x là hoành độ 1<=x<=80, y là tung độ 1<=y<=25.

CLRSCR: xoá toàn bộ màn hình, sau khi xoá con trỏ có toạ độ (1,1)

TEXTCOLOR(màu): thiết lập màu cho các ký tự văn bản, Turbo Pascal 7.0 cho phép lựa chọn tối đa là 15 màu. Các màu này được đánh số từ 0 đến 15 hoặc có thể viết trực tiếp bằng tiếng Anh ví dụ: Black (đen), Blue (xanh da trời), Red (đỏ), Green (xanh lá mạ), Yellow (vàng), Mangeta (tím), Brown (nâu),.....

TEXTBACKGROUND(màu): chọn màu nền cho văn bản

Hai thủ tục TEXTCOLOR(màu) và TEXTBACKGROUND(màu) sẽ có tác động cho đến khi gặp một thủ tục chọn màu mới.

Chương 4

CÁC CẤU TRÚC LẬP TRÌNH

Như đã nêu Pascal là một ngôn ngữ lập trình có cấu trúc, tính "Cấu trúc" của ngôn ngữ thể hiện ở hai khía cạnh: Dữ liệu có cấu trúc và lập trình có cấu trúc. Các loại dữ liệu có cấu trúc sẽ được đề cập trong các chương tiếp theo. Khía cạnh lập trình có cấu trúc thể hiện ở chỗ ngoài các lệnh đơn giản Pascal còn có các lệnh có cấu trúc mà ta quen gọi là cấu trúc lập trình, các lệnh này có thể nhóm thành các khối (block) mỗi khối đặt giữa một cặp từ khoá Begin và End; mỗi block có thể thực hiện một công việc cụ thể nào đó. Xét rộng hơn, một chương trình lớn có thể bao gồm nhiều khối lớn mà ta gọi là Modun, các Modun thường được tách khỏi chương trình mẹ để tạo nên các chương trình con. Nhiệm vụ quan trọng đầu tiên của người lập trình chính là việc thiết kế cấu trúc chương trình mà điều này không thể thực hiện được nếu không nắm vững các cấu trúc lập trình.

I. CÂU LỆNH KIỂM TRA

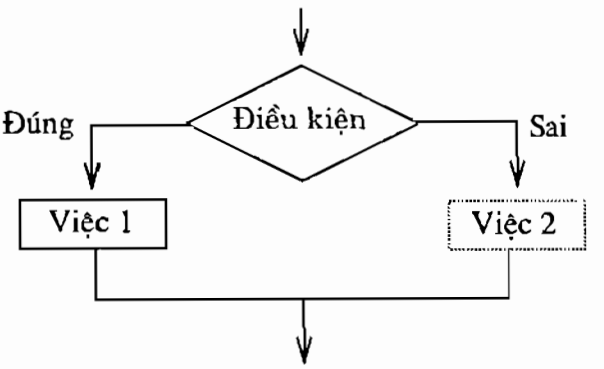
Cú pháp:

IF <điều kiện> THEN

<việc 1>

[ELSE]

[<việc 2>]

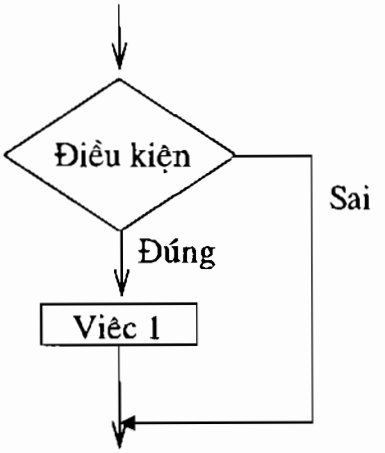


Hình 4.1 Sơ đồ khối cấu trúc kiểm tra

<Điều kiện> trong câu lệnh IF là một biểu thức kiểu Boolean, nghĩa là biểu thức này chỉ nhận một trong hai giá trị TRUE hoặc FALSE.

Câu lệnh được thực hiện như sau: Gặp từ khóa IF, Pascal sẽ kiểm tra điều kiện, nếu <điều kiện> là TRUE thì tiến hành làm <việc 1>, còn nếu <điều kiện> là FALSE thì làm <việc 2>.

Trong câu lệnh IF mệnh đề ELSE là không bắt buộc phải có. Khi không có ELSE thì cũng không có <việc 2>, nếu <điều kiện> là TRUE thì làm <việc 1> còn nếu là False thì không làm mà chuyển đến lệnh kế tiếp ngay sau <việc 1>, trường hợp <việc 1> là một lệnh ghép thì chuyển đến lệnh sau từ khoá End; của lệnh ghép đó (hình 4.2).



Hình 4.2

<Việc 1> và <việc 2> có thể là một câu lệnh đơn lẻ hoặc là một tập hợp các câu lệnh (lệnh ghép). Nếu cả <việc 1> và <việc 2> đều là lệnh đơn thì giữa từ khoá IF và từ khoá ELSE không có dấu ngăn cách lệnh ";" (xem ví dụ .4.1)

Ví dụ 4.1

```
Program Vd41;
Var lam:char;
Begin
Lam:='k';
IF lam='k' THEN
Writeln('Ket thuc cong viec - tro ve man hinh chinh')
ELSE
Writeln('Hay chon cong viec tiep theo');
Readln;
End.
```

Nếu <việc 1> hoặc <việc 2> là lệnh ghép thì phải đặt chúng giữa các từ khoá BEGIN và END. Từ khoá END của nhóm lệnh ghép thứ nhất đứng trước ELSE không có dấu ngăn cách câu lệnh ";", từ khoá End của nhóm lệnh 2 <viec 2> báo kết thúc câu lệnh IF nên phải có dấu ";' (ví dụ 4.2).

Ví dụ 4.2:

```
Program   Cautruc_if;
Uses crt;
Var  a,b,x,f1,f2:real;     ham:byte;
  Begin
  Clrscr;
  Write('Chon ham F1 (1) hay F2 (2) ?'); readln(ham);
  if ham =1 then
```

```
begin
   x:=5;   a:=1.5;  b:=2;
   f1:=a*x+b;
end
else
begin
   x:=-5;   a:=-1.5;  b:=-2;
   f2:=a/x + b;
end;
if ham=1 then
    Writeln('Gia tri ham F1 tinh duoc la ', f1:8:2)
else
     Writeln('Gia tri ham F2 tinh duoc la ', f2:8:2);
readln;
End.
```

Ví dụ 4.2 ứng dụng cấu trúc IF trong cả hai trường hợp, đầu tiên trên màn hình xuất hiện câu hỏi:

Chon ham F1 (1) hay F2 (2)?

Nếu người sử dụng muốn tính hàm F1 (f1 = ax + b) thì gõ số 1, lúc này chương trình sẽ thực hiện phép gán các giá trị dương cho x, a, b sau đó tính hàm F1. Tổng số lệnh là 4 do đó chúng phải đặt giữa các từ khoá Begin và End. Còn nếu người sử dụng gõ số 2 nghĩa là tính hàm F2 (thực ra muốn tính hàm F2 người sử dụng có thể gõ một phím bất kỳ khác số 1). Nhóm lệnh tính hàm F2 cũng tương đương như tính hàm F1 do đó cũng phải đặt giữa Begin và End, chỉ có điều khác là từ khoá End ở đây dùng để kết thúc cấu trúc IF cho nên phải có dấu " ; ". Đoạn sau của chương trình dùng để viết kết quả tính toán ra màn hình, mỗi nhánh của cấu trúc IF chỉ có một lệnh nên không cần đến từ khoá Begin và End.

Bài toán kinh điển minh hoạ cho việc dùng cấu trúc IF là lập trình giải phương trình bậc hai $ax^2 + bx + c = 0$. Thuật giải của chương trình đã biết trong chương trình phổ thông. Khi chưa quen với việc ghép các cấu trúc IF lồng nhau chúng ta có thể sử dụng chúng một cách riêng rẽ như ví dụ 4.3:

Ví dụ 4.3

```
Program Giai_PTBH;
uses crt;
Var
a,b,c,delta:Real
Begin
clsscr;
gotoxy(30,10);
Writeln(' GIAI PHUONG TRINH BAC HAI a*x*x + b*x + c = 0 ');
Writeln(' Xin moi nhap cac he so a, b, c ');
```

```
Write(' a = '); Readln(a);
Write(' b = '); Readln(b);
Write(' c = '); Readln(c);
Delta:=b*b-4*a*c;
   if delta < 0 then   Writeln(' Phuong trinh khong co nghiem thuc')
   if delta = 0 then
      Begin
         Writeln('Phuong trinh co nghiem kep ');
         Writeln('x1 = x2 = ',-b/(2*a):8:2);
      End;
      If delta > 0 then
      Begin
         Writeln('Phuong trinh co hai nghiem phan biet ');
         Writeln('x1 = ', (-b+sqrt(delta))/(2*a));
         Writeln('x2 = ', (-b-sqrt(delta))/(2*a));
      end;
Readln;
End.
```

Cấu trúc IF ... có thể sử dụng lồng vào nhau thành nhiều tầng, điều kiện duy nhất là cấu trúc IF con phải nằm gọn trong lòng cấu trúc mẹ, không được chéo nhau (ví dụ 4.4)

```
IF <điều kiện 1> then
  Begin
     IF <điều kiện 2> then
       Begin
         ......
       End;
  End
  ELSE
  ..........
```

Ví dụ 4.4

```
Program  Giai_PTBH;
Uses crt;
Var  a,b,c,delta:Real
Begin
clsscr;  gotoxy(30,10);
Writeln(' GIAI PHUONG TRINH BAC HAI a*x*x + b*x + c = 0 ');
Writeln(' Xin moi nhap cac he so a, b, c ');
```

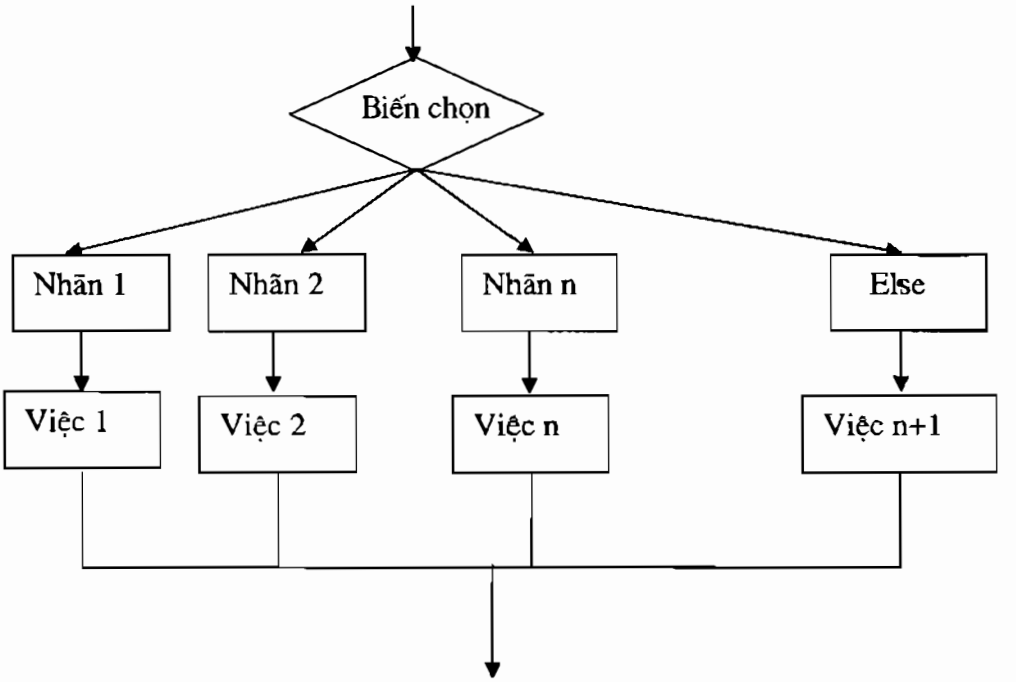
```
Write(' a = '); Readln(a);
Write(' b = '); Readln(b);
Write(' c = '); Readln(c);
Delta:=b*b-4*a*c;
if delta < 0 then
   Writeln(' Phuong trinh khong co nghiem thuc')
   else
     if delta = 0 then
      Begin
         Writeln('Phuong trinh co nghiem kep ');
         Writeln('x1 = x2 = ',-b/(2*a):8:2);
      End
      else
      Begin
         Writeln('Phuong trinh co hai nghiem phan biet ');
         Writeln('x1 = ', (-b+sqrt(delta))/(2*a));
         Writeln('x2 = ', (-b-sqrt(delta))/(2*a));
      end;
Readln;
End.
```

II. CẤU TRÚC RẼ NHÁNH

Cấu trúc IF... chỉ có thể lựa chọn một trong hai nhánh ứng với các giá trị TRUE hoặc FALSE của biểu thức <điều kiện>. Trong trường hợp ta phải lựa chọn nhiều khả năng thì phải dùng cấu trúc rẽ nhánh CASE.. OF... (hình 4.3).

Dạng đơn giản:	Dạng tổng quát:
CASE <Biến chọn> OF Nhãn1: <lệnh1>; Nhãn 2: <lệnh2>; ..... Nhãn n: <lệnhn>; END;	CASE <biến chọn> OF Nhãn 1: <lệnh1>; Nhãn 2: <lệnh2>; ..... Nhãn n: <lệnhn> ELSE <lệnh n+1>; END;

Câu lệnh CASE bao gồm một **<biến chọn>** tiếp theo là danh sách các lệnh, mỗi lệnh đặt sau một nhãn. **Nhãn** ở đây được hiểu là một giá trị hoặc một khoảng giá trị mà biến chọn sẽ nhận, nếu nhãn là một khoảng giá trị thì ta viết giá trị đầu tiếp đến hai dấu chấm và giá trị cuối.



Hình 4.3

* **<Biến chọn>** sau từ khoá CASE có thể là biểu thức toán, biểu thức logic ... nhưng nó phải cho những giá trị cụ thể, các giá trị này có thể thuộc kiểu dữ liệu ký tự hoặc số , nếu là số thì phải là kiểu số nguyên.

* <Việc 1>, <việc 2>.... <việc n> có thể là những lệnh đơn lẻ hay lệnh ghép giống như trường hợp lệnh IF... THEN.....

Chú ý:

Lệnh rẽ nhánh CASE .. OF.. bao giờ cũng bắt đầu bằng từ khoá CASE và kết thúc bằng từ khoá End;

Hoạt động của cấu trúc:

* Nếu giá trị hiện thời của <Biến chọn> bằng một nhãn nào đó hoặc thuộc vào một khoảng giá trị ghi sau một Nhãn nào đó thì câu lệnh viết sau nhãn này sẽ được thực hiện. Nếu không có nhãn nào chứa giá trị hiện thời của <biến chọn> thì thực hiện các lệnh viết sau từ khoá ELSE tức là <việc n>, nếu không có ELSE thì bỏ qua các lệnh của cấu trúc chuyển xuống thực hiện các lệnh sau từ khoá End của chương trình.

Ví dụ: Cho biết mã một số phím cơ bản trên bàn phím (xem bảng) khi người sử dụng gõ một phím nào đó, màn hình sẽ hiện lên thông báo tên phím mà người đó đã gõ.

Tên phím	Esc	Enter	Tab	BackSpace
Mã phím	27	13	9	8

Ví dụ 4.5:

```
Program bay_phim;
Uses crt;
Var chu:char;
 Begin
 Clrscr;
 Writeln('Xin moi bam mot trong cac phim: Esc, Tab, Enter, Backspace ');
 chu:=readkey;
   Case chu of
    #13: Write('Ban vua go phim Enter');
    #27: Write('Ban vua go phim Esc');
    #9: Write('Ban vua go phim Tab');
    #8: Write('Ban vua go phim BackSpace');
   end;
   readln;
 End.
```

Ví dụ 4.6: Đổi năm sinh dương lịch sang năm sinh âm lịch

Theo âm lịch tên của năm được viết bởi hai đại lượng là Can và Chi. Hàng chi tượng trưng cho mười hai con giáp là:

Tý, Sửu, Dần, Mão, Thìn, Tỵ, Ngọ, Mùi, Thân, Dậu, Tuất, Hợi.

Hàng Can có mười giá trị là:

Giáp, Ất, Bính, Đinh, Mậu, Kỷ, Canh, Tân, Nhâm, Quý

Ghép lần lượt Can và Chi ta được tên năm. Do Can và Chi có số lượng không giống nhau nên sau mười lần ghép thì hàng can sẽ lặp lại còn hàng Chi phải sau 12 lần ghép mới lặp lại. Bội số chung nhỏ nhất của 10 và 12 là 60 do đó sau 60 năm thì tên năm lại lặp lại như cũ.

Nhận xét rằng nếu lấy năm sinh dương lịch (viết đủ bốn số) chia cho 12 thì số dư sẽ của phép chia sẽ rơi vào một trong các giá trị: 0, 1, 2, 3, 4,5, 6, 7, 8, 9, 10, 11. Còn lấy năm sinh chia cho 10 thì số dư sẽ nằm trong các giá trị: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Như đã biết phép chia hai số nguyên để lấy số dư là phép MOD.

Giả thiết ký hiệu năm sinh là NS chúng ta có được bảng đối chiếu sau đây:

Ns mod 12	0	1	2	3	4	5	6	7	8	9	10	11
Chi	Thân	Dậu	Tuất	Hợi	Tí	Sửu	Dần	Mão	Thìn	Tỵ	Ngọ	Mùi
Ns mod 10	0	1	2	3	4	5	6	7	8	9		
Can	Canh	Tân	Nhâm	Quý	Giáp	Ất	Bính	Đinh	Mậu	Kỷ		

Bảng 4.1

Ví dụ 4.6

```
Program  Nam_am_lich;
Uses crt;
```

```
Var ns:Integer;
  can,chi:string[4];
Begin
clrscr;
  Write('Hay cho biet nam sinh cua ban ');
  Readln(ns);
  Case ns mod 12 of
       0: chi:='Than';
       1: chi:='Dau';
       2: chi:='Tuat';
       3: chi:='Hoi';
       4: chi:='Ti';
       5: chi:='Suu';
       6: chi:='Dan';
       7: chi:='Mao';
       8: chi:='Thin';
       9: chi:='Ty';
       10: chi:='Ngo';
       11: chi:='Mui';
   End;
  Case ns mod 10 of
       0: can:= 'Canh';
       1: can:= 'Tan';
       2: can:= 'Nham';
       3: can:= 'Quy';
       4: can:= 'Giap';
       5: can:= 'At';
       6: can:= 'Binh';
       7: can:= 'Dinh';
       8: can:= 'Mau';
       9: can:=  'Ky';
    End;
   Writeln('Ban sinh nam ',can,'  ',chi);
  readln;
  end.
```

Chương trình trong ví dụ 4.6 chọn biến NS cho năm sinh, chữ "Ti" cho Tí (chuột) và chữ "Ty" cho Tỵ (Rắn).

Chạy chương trình và gõ vào năm sinh ví dụ 2000, chương trình sẽ thông báo bạn sinh năm Canh Thìn.

Có thể phát triển chương trình trên thêm một chút cho vui nếu bạn quan tâm đến Dịch học. Với nguyên tắc: " Niên vi cốt, Nguyệt vi bì" nghĩa là lấy năm sinh làm xương, tháng sinh làm da. Da bọc kín xương thì yên ấm, vui vẻ, ngược lại thì bị lạnh, bị khó khăn.

Chương trình sẽ yêu cầu cho biết thêm tháng sinh. Ví dụ năm sinh là năm Sửu (Trâu), tháng sinh là tháng Mão (Mèo), như vậy da mèo không thể bọc kín xương trâu. Khi đó chương trình sẽ hiện một câu thông báo nào đó tuỳ các bạn thiết kế.

III. CẤU TRÚC LẶP

Pascal đã thiết kế hai kiểu lặp là:

* Lặp với số lần lặp đã xác định

* Lặp với số lần lặp chưa xác định

Mỗi kiểu lặp nêu trên lại có hai cấu trúc kèm theo như vậy chúng ta có cả thảy bốn cấu trúc lặp.

1. Câu lệnh lặp có số lần lặp xác định trước FOR....

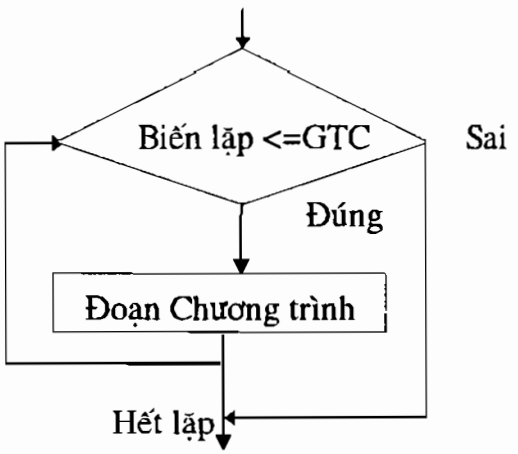
Vòng lặp có số lần lặp xác định có hai mẫu tổng quát là:

a. FOR biến lặp := giá trị đầu TO giá trị cuối DO

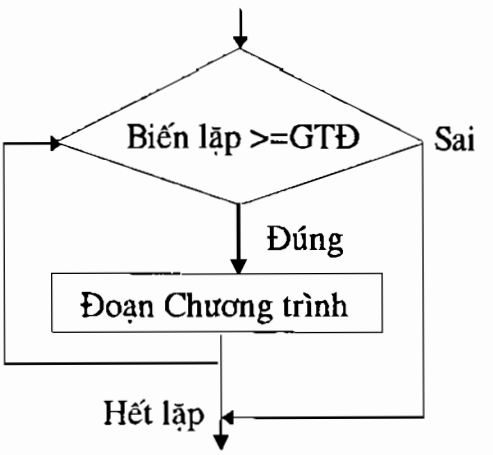
<công việc > (hình 4.4).

b. FOR biến lặp := giá trị cuối DOWNTO giá trị đầu DO

<công việc> (hình 4.5).



Hình 4.4



Hình 4.5

Trong hai mẫu lệnh trên **biến điều khiển, giá trị đầu** và **giá trị cuối** phải cùng một kiểu dữ liệu, chúng phải thuộc kiểu dữ liệu đếm được như kiểu số nguyên hoặc kiểu ký tự.

Công việc sau từ khoá DO có thể là một câu lệnh hay một lệnh ghép nhưng không được chứa lệnh gán giá trị cho biến điều khiển nghĩa là không được chứa các lệnh làm thay đổi giá trị của Biến điều khiển.

Ví dụ 4.7 trình bày cách vẽ một hình chữ nhật trong đó hai cạnh nằm ngang vẽ bằng nét "-" còn hai cạnh thẳng đứng vẽ bằng dấu "*"

Ví dụ 4.7:

```
Program LENH_FOR;
USES CRT;
Var
i,j:integer;
BEGIN
   clrscr;
   gotoxy(1,8);
   for i:=1 to 76 do write('_');   (* Viết 76 nét gạch ngang ở dòng thứ 8*)
   gotoxy(1,12);
   for i:=76 downto 1 do write('_');
   gotoxy(1,9);
   for j:=4 downto 1 do writeln('*');
   gotoxy(76,9);  write('*');
   gotoxy(76,10);  write('*');
   gotoxy(76,11);  write('*');
   gotoxy(76,12);  write('*');
   repeat until keypressed;
END.
```

Ví dụ 4.8 trình bày một ứng dụng khác của lệnh lặp FOR, ở đây đã dùng hai vòng lặp lồng nhau để vẽ một khung hình chữ nhật bằng ký tự số 178 trong bảng mã ASCII (Muốn làm xuất hiện ký tự này ta đè phím Alt và gõ con số 178). Trong chương trình đã sử dụng hai thủ tục chọn màu là:

Textcolor(mau): chọn màu cho ký tự sẽ viết ra màn hình,

Textbackground(mau): chọn màu nền cho văn bản

(mau) là số hiệu màu đã được quy định sẵn, trong Pascal có thể chọn mã màu theo các số từ 0 đến 15 hoặc viết trực tiếp tên màu bằng tiếng Anh (red, blue, green..)

* Thủ tục Random(15) là một thủ tục chuẩn cho kết quả là một số ngẫu nhiên n (kiểu Integer) giá trị của n nằm trong khoảng 0 <= n <= 15

Ví dụ 4.8:

```
Program VE_KHUNG;
uses crt;
var
n,i,j:integer;
Begin
```

```
clrscr;
n:=2;
for j:=1 to 20 do
begin
textcolor(random(15));  (* Chọn ngẫu nhiên màu các ký tự từ 0-15 *)
textbackground(blue);   (* Chọn màu nền là màu xanh da trời *)
gotoxy(15,n);
for i:=1 to 50 do write(chr(178));
n:=n+1;
End;
repeat until keypressed;
End.
```

2. Vòng lặp có số bước lặp không xác định

Tồn tại hai kiểu vòng lặp với số bước lặp không xác định là:

REPEAT
  <Công việc>
UNTIL
  <biểu thức Boolean>
	WHILE <biểu thức Logic> DO
	  Begin
	  <công việc>
	  End;

Trong vòng lặp thứ nhất REPEAT...UNTIL, máy sẽ thực hiện công việc tức là các lệnh giữa Repeat và Until sau đó kiểm tra <biểu thức Boolean>, nếu biểu thức nhận giá trị TRUE thì lặp lại <công việc> nếu FALSE thì kết thúc vòng lặp và thực hiện các lệnh sau Until.

Trong vòng lặp thứ hai WHLIE....DO, máy kiểm tra <biểu thức Boolean> nếu TRUE thì thực hiện công việc giữa Begin và End, nếu FALSE thì thoát khỏi vòng lặp và thực hiện các lệnh sau End.

Sự khác nhau cơ bản giữa hai vòng lặp này là: vòng lặp REPEAT..UNTIL máy thực hiện công việc trước rồi kiểm tra điều kiện sau như vậy máy ít nhất cũng thực hiện vòng lặp một lần, còn vòng lặp WHILE..DO thì kiểm tra điều kiện trước do đó có thể không thực hiện <công việc> một lần nào.

Chú ý: *Đây là vòng lặp có số bước lặp không xác định nên cần phải có một biến điều khiển để tạo ra giá trị FALSE cho biểu thức Boolean, nếu không vòng lặp sẽ không dừng lại và chúng ta buộc phải tắt máy để sửa chữa chương trình.*

Ví dụ 4.9:

Tính giá trị sin(pi/n) trong đó pi là giá trị ngầm định của số PI đã có trong Turbo Pascal, còn n là một số nguyên lấy giá trị từ 1 đến 20. Kết quả ghi ra màn hình bắt đầu từ vị trí cột 30.

```
Program  tinhsin(x);
uses crt;
Var
x:real;  n:integer;
Begin
clrscr;
n:=1;  x:=pi;
while n<=20 do
begin
gotoxy(30,n);
write('sin(pi/',n,') = ',sin(x/n):10:7);
n:=n+1;
end;
gotoxy(30,24);
Repeat until keypressed;
End.
```

Kết quả sẽ hiện trên màn hình như sau:
sin(pi/1) = 0.0000000 sin(pi/2) = 1.0000000 sin(pi/3) = 0.8660254 sin(pi/4) = 0.7071068
..... sin(pi/20) = 0.1564345

Có thể sửa chữa chương trình bằng cách bố trí các kết quả trên từng dòng hoặc đưa thêm vào lệnh đọc giá trị n và lệnh lựa chọn làm tiếp hay thôi... Xin mời độc giả tự làm.

3. Lệnh nhảy vô điều kiện GOTO <địa chỉ>

Lệnh nhảy vô điều kiện GOTO <địa chỉ> cho phép nhảy từ vị trí của câu lệnh Goto đến câu lệnh có địa chỉ viết ở phía trước, sau <địa chỉ> là dấu ":" rồi đến các từ ngữ của câu lệnh. Muốn sử dụng lệnh Goto phải dùng từ khoá LABEL để khai báo ở phần đầu chương trình các nhãn sẽ sử dụng làm địa chỉ, nhãn có thể là một số nguyên hoặc một tên được viết cách nhau bởi dấu phảy sau từ khoá Label.

Ví dụ 4.10:

```
Program  Lenh_GOTO;
uses crt;
Label 1,2;
Const a=12345;
Var
lam:char;   matma, n, i, j:integer ;
Begin
     clrscr;   n:=14;
     textbackground(white);  textcolor(red);
     for j:=1 to 3 do
     Begin
     gotoxy(20,n);
     for i:=1 to 40 do write(chr(178));
     n:=n+1;
```

```
end;
gotoxy(23,15); Write(' Hay cho biet mat ma cua ban? ');
textbackground(red);
readln(matma);
textbackground(white);
if matma=a then goto 2 else goto 1;
1: if matma<>a then
Begin
gotoxy(20,18);
write(' Sai mat ma, hay khai bao mot lan nua ');
gotoxy(60,18); textbackground(red);
readln(matma); textbackground(white);
clrscr;
textcolor(red);
n:=14;
textbackground(white);  textcolor(9);
for j:=1 to 3 do
begin
     gotoxy(20,n);
     for i:=1 to 46 do write(chr(178));
     n:=n+1;
end;
textcolor(red);
gotoxy(22,15); Writeln(' Xin loi, Ban khong duoc phep tiep tuc!! ');
textcolor(blue);
gotoxy(30,18); write('Hay bam Esc de ve DOS ');
textbackground(white);
lam:=readkey;
if lam=#27 then
clrscr;
halt;
End;
2: clrscr; n:=11;
textbackground(white);  textcolor(red);
for j:=1 to 3 do
begin
     gotoxy(20,n);
     for i:=1 to 40 do write(chr(178));
     n:=n+1;
end;
gotoxy(22,12); Write(' Xin chao, moi ban chon cong viec ? ');
n:=15;
for j:=1 to 7 do
begin
     gotoxy(38,n); textcolor(9);
     for i:=1 to 28 do write(chr(178));
     n:=n+1;
end;
```

```
    textcolor(0);
    gotoxy(40,16); Write(' 1. Hang khong dan dung ');
    gotoxy(40,17); Write(' 2. Dau khi ');
    gotoxy(40,18); Write(' 3. Giao vien ');
    gotoxy(40,19); Write(' 4. Thay thuoc ');
    gotoxy(40,20); Write(' 5. Xa vien hop tac xa ');
    repeat until keypressed;
    textbackground(blue);
    textcolor(15);
    clrscr;
End.
```

Ví dụ 4.10 trình bày cách tạo mật mã chương trình trong đó có sử dụng lệnh nhảy vô điều kiện GOTO. Tuy nhiên chúng ta có thể không dùng lệnh GOTO mà dùng lệnh CASE..OF. Nói chung theo kinh nghiệm nên tránh dùng lệnh GOTO. Trong trường hợp cần dùng thì phải lưu ý rằng không dùng lệnh GOTO để nhảy từ ngoài vào trong một chương trình con (CTC) mà chỉ có thể nhảy từ trong CTC ra ngoài.

Chú ý: *Để bảo đảm bí mật khi người sử dụng bấm các ký tự mật mã ta cần quy định màu của ký tự và màu nền là như nhau.*

Ví dụ 4.10 là một chương trình chưa hoàn chỉnh vì trong trường hợp bạn đã chọn một công việc (sau nhãn 2 của lệnh nhảy GOTO) thì cần các quá trình xử lý tiếp theo tức là xây dựng các chương trình con ứng với mỗi công việc. Chúng ta sẽ tiếp tục điều này ở các chương sau.

BÀI TẬP ỨNG DỤNG CHƯƠNG 4

1. Cho 100 con trâu ăn 100 bó cỏ, trong đó trâu đứng mỗi con được ăn 5 bó, trâu nằm mỗi con 3 bó còn trâu già cứ 3 con được ăn 1 bó. Lập chương trình để tìm xem mỗi loại trâu có bao nhiêu con?
2. Lập chương trình để in ra màn hình các ký tự của bảng mã ASCII từ ký tự số 32 đến ký tự số 255.
3. Lập chương trình để chuyển một số từ hệ thập phân sang hệ nhị phân. Đưa kết quả ra màn hình.
4. Gọi các cạnh của tam giác là a, b, c. Đặt p = (a+b+c)/2. Diện tích tam giác được tính theo công thức:

   $$S = p*(p-a)*(p-b)*(p-c)$$

   Lập chương trình nhập các số a,b,c và kiểm tra xem chúng có tạo thành một tam giác hay không. Nếu có hãy tính diện tích tam giác, đưa kết quả ra màn hình.
5. Cho một số có 3 chữ số abc, tìm a, b, c sao cho

   $$abc = a*a*a + b*b*b + c*c*c$$

   (Nghiệm bài toán là: 000, 001, 153, 370, 371, 407)
6. Lập chương trình tính diện tích tam giác biết hai cạnh là a, b, góc xen giữa hai cạnh là anpha.
7. Cho một khối trụ bán kính đáy là r, chiều cao là h. Lập chương trình tính thể tích và diện tích toàn phần của khối trụ đó.
8. Một thửa ruộng hình chữ nhật kích thước 120 x 145 (mét). Tính sản lượng thóc thu được nếu một năm cấy hai vụ, năng suất vụ mùa 180 kg/sào, năng suất vụ chiêm bằng 85% vụ mùa.
9. Tốc độ tăng dân số của một quốc gia là 2% một năm. Năm 1990 dân số nước này là 65 triệu người. Hỏi năm 2000 sẽ có bao nhiêu người.
10. Tính bình quân mỗi người một tháng ăn hết 12 kg gạo. Hãy lấy các số liệu trong bài tập 9 để tính lượng gạo cần có trong các năm từ 1990 đến 2000.

Chương 5

DỮ LIỆU KIỂU MẢNG

I. KHÁI NIỆM CHUNG

Trong chương 2 chúng ta đã nghiên cứu các kiểu dữ liệu đơn giản bao gồm Real, Integer, Boolean, Char và kiểu dữ liệu do người lập trình định nghĩa (TYPE). Từ các kiểu dữ liệu này Turbo Pascal xây dựng nên một số kiểu dữ liệu có cấu trúc là kiểu mảng (Array), kiểu tập (Set), kiểu bản ghi (Record), kiểu tệp (File). Mỗi kiểu dữ liệu có cấu trúc được đặc trưng bởi kiểu của các phần tử cấu tạo nên chúng. Phương pháp truy nhập vào một phần tử nào đó của kiểu có cấu trúc tuỳ thuộc vào cách thức mà nó được tạo thành.

II. KIỂU MẢNG (ARRAY)

Một mảng dữ liệu bao gồm một số hữu hạn các phần tử có cùng kiểu dữ liệu. Người lập trình có thể định nghĩa trước kiểu dữ liệu thông qua từ khoá TYPE rồi mới định nghĩa mảng. Nếu sử dụng các kiểu đơn giản chuẩn đã có thì có thể định nghĩa trực tiếp kiểu mảng. Việc khai báo biến kiểu mảng viết sau từ khoá VAR như các biến khác và phải bao gồm:

- Tên mảng
- Từ khoá ARRAY
- Chỉ số mảng đặt giữa hai dấu ngoặc vuông.
- Từ khoá OF
- Kiểu của các thành phần của mảng (còn gọi là kiểu phần tử)

Ghi chú:

Chỉ số của mảng chỉ có thể là kiểu đơn giản (kiểu đếm được) và có thể nhận một trong các kiểu dữ liệu sau đây:

- Kiểu ký tự
- Kiểu tập con (của Integer hoặc Char)
- Kiểu do người lập trình tự định nghĩa
- Kiểu Boolean

Kiểu của chỉ số không được là kiểu Real hoặc một kiểu có cấu trúc khác.

Ví dụ 5.1: Định nghĩa kiểu dữ liệu và kiểu biến mảng

```
Type
Ngay = (mot,hai,ba,bon,nam,sau,bay,);
```

```
Var
giolam: array[1..8] of integer;
Ngay_trong_tuan: array[ngay] of string[10];
```

Trong ví dụ 5.1 ta đã định nghĩa một kiểu dữ liệu mới là kiểu "ngay" kiểu này được tạo nên bởi 7 thành phần có tên là mot, hai, ba, bon, nam, sau, bay, . Tiếp đó ta đã định nghĩa hai biến mảng là giolam và ngay_trong_tuan.

Mảng giolam là mảng gồm 8 phần tử, các phần tử của giolam có kiểu là Integer, mảng ngay_trong_tuan có 7 phần tử, các phần tử của ngay_trong-tuan có chỉ số là mot, hai, ba... bay , các phần tử này nhận kiểu dữ liệu là kiểu String (kiểu chuỗi ký tự) với độ dài tối đa là 10 ký tự (Kiểu string sẽ giới thiệu trong chương sau).

Sau khi định nghĩa mảng, chúng ta có thể gán giá trị cho các phần tử của mảng từ bàn phím, thông thường nên viết một chương trình để nhập dữ liệu như trong ví dụ sau:

Ví dụ 5.2:

```
Program  mang;
uses crt;
type  ngay = (mot,hai,ba,bon,nam,sau,bay);
var  gio_lam:array[1..8] of integer;        i:integer;
     ngay_trong_tuan: array[ngay] of string[10];
Begin
clrscr;
for i:=1 to 12 do
Begin
   writeln('gan du lieu cho mang');
   write('thang[',i,'] = '); readln(thang[i]);
End;
clrscr;
Writeln('Gan ten ngay cho mang ngay_trong_tuan: ');
Write('Ngay dau tuan la: '); readln(ngay_trong_tuan[mot]);
Write('Ngay thu hai la: '); readln(ngay_trong_tuan[hai]);
Write('Ngay thu ba la: '); readln(ngay_trong_tuan[ba]);
Write('Ngay thu tu la: '); readln(ngay_trong_tuan[bon]);
Write('Ngay thu nam la: '); readln(ngay_trong_tuan[nam]);
Write('Ngay thu sau la: '); readln(ngay_trong_tuan[sau]);
Write('Ngay thu bay la: '); readln(ngay_trong_tuan[bay]);
clrscr;
writeln('Ten cac ngay trong tuan la: ');
writeln(ngay_trong_tuan[mot]);
writeln(ngay_trong_tuan[hai]);
writeln(ngay_trong_tuan[ba]);
writeln(ngay_trong_tuan[bon]);
writeln(ngay_trong_tuan[nam]);
writeln(ngay_trong_tuan[sau]);
writeln(ngay_trong_tuan[bay]);
repeat until keypressed;
end.
```

Ví dụ 5.2 là một chương trình phát triển từ ví dụ 5.1, mảng ngay_trong_tuan có kiểu chỉ số là kiểu ngay và kiểu dữ liệu là kiểu chuỗi ký tự.

Chạy chương trình trên ta có màn hình gán dữ liệu và kết quả viết các phần tử mảng ra màn hình như sau:

```
Ten cac ngay trong tuan la :
Thu Hai
Thu Ba
Thu tu
Thu Nam
Thu Sau
Thu Bay
Chu nhat
```

Với mảng đã định nghĩa ta có thể truy nhập trực tiếp vào các phần tử của mảng như lệnh viết các phần tử của mảng ra màn hình trong ví dụ 5.2 hoặc thực hiện các phép toán lên các phần tử, ví dụ:

```
Writeln(ngay_trong_tuan[nam]+ngay_trong_tuan[ba])
```

Kết quả hiện trên màn hình là:

```
Thu namThu ba
```

Ví dụ 5.3: Lập chương trình nhập vào mảng m1 n phần tử và m2 m phần tử. Ghép hai mảng m1 và m2 thành mảng m3, sắp xếp m3 theo thứ tự tăng dần và viết kết quả ra màn hình.

```
Program  ghepmang;
uses crt;
var
   m1 : array[1..100] of real;
   m2 : array[1..100] of real;
   m3 : array[1..200] of real;
   n,m,i,j:integer; btg: real;
begin
clrscr;
Write(' Cho biet so phan tu cua mang m1 va m2 ');readln(n,m);
for i:=1 to n do
begin
   write('m1[',i,'] = '); readln(m1[i]);  (* nhập số liệu vào mảng m1 *)
end;
writeln;
for i:=1 to m do
begin
  write('m2[',i,'] = '); readln(m2[i]);  (* nhập dữ liệu vào mảng m2*)
end;
for i:=1 to n do m3[i]:=m1[i];          (*nối m1 vào m3*)
for j:=1 to m do m3[j+n]:=m2[j];        (*nối m2 vào m3*)
for i:=1 to n+m-1 do                    (*sắp xếp m3 theo chiều tăng dần*)
for j:=i+1 to n+m do
```

```
if m3[i]>m3[j] then
begin
  btg:=m3[j];m3[j]:=m3[i];m3[i]:=btg;
end;
writeln; textcolor(14); textbackground(green);
writeln('Thu tu cua mang ghep m3 sau khi sap xep : ');
for i:=1 to n+m do write(m3[i]:5:1,' ');     (* viết m3 ra màn hình*)
writeln;
 readln;
 end.
```

III. MẢNG NHIỀU CHIỀU

Khái niệm về mảng cũng tương tự như khái niệm về ma trận trong toán học. Chúng ta đã từng biết đến các khái niệm ma trận hàng, ma trận cột và ma trận vuông (hoặc chữ nhật). Những mảng định nghĩa trong mục II thuộc loại mảng một chiều. Như đã nêu ở trên kiểu dữ liệu của chỉ số bị hạn chế là kiểu đơn giản, còn kiểu dữ liệu của các thành phần mảng không bị hạn chế, như vậy chúng ta hoàn toàn có thể chọn kiểu dữ liệu có cấu trúc cho các thành phần mảng, trong trường hợp đó ta có mảng nhiều chiều.

Ví dụ :

```
Var
Mang_vuong:array[1..3] of array[1..3] of Integer;
Mangcn: array[1..5] of array[1..7] of char;
....
```

Với cách khai báo trên Mang_vuong là một biến kiểu mảng gồm 3 thành phần Mang_vuong[1], Mang_vuong[2] và Mang_vuong[3]. Mỗi thành phần lại là một mảng của ba số nguyên. Mang_vuong thực chất là một ma trận vuông ba hàng và ba cột.

Mangcn là một ma trận chữ nhật 5 hàng, 7 cột chứa các ký tự của bảng mã ASCII.

Khái niệm mảng được dùng rộng rãi trong các phép tính với ma trận, chúng ta sẽ xét việc nhân hai ma trận chữ nhật qua việc sử dụng biến mảng. Điều kiện để nhân hai ma trận A và B là số cột của ma trận A phải bằng số hàng của ma trận B. Xét hai ma trận A[m.n] (m hàng n cột) và ma trận B[n.r] (n hàng r cột) . Khi đó ma trận tích C = A*B sẽ là một ma trận có m hàng và r cột. Mỗi phần tử của ma trận tích Cij được tính theo công thức:

$$C_{ij} = \sum_{k=1}^{r} A_{ik} * B_{kj}$$

Ví dụ 5.4

```
Program Nhan_ma_tran;
Uses crt;
Const  m=3; r=4; n=5;
Var
   A:array[1..3] of array[1..5] of Integer;
   B:array[1..5] of array[1..4] of Integer;
   C:array[1..5] of array[1..4] of Integer;
   i,j,k: Integer;
```

```
Begin
CLRSCR;
writeln('CHUONG TRINH NHAN HAI MA TRAN A*B = C ');
Writeln('GAN GIA TRI CHO MANG A ( 3 hang - 5 cot ) ');
for i:=1 to m do    (*Chi so hang tu 1-3*)
Begin
for j:=1 to n do      (*Chi so cot tu 1-5*)
Begin
Write('A[',i,',',j,'] = '); read(A[i,j]);
End;
writeln;
End;
Writeln('GAN GIA TRI CHO MANG B ( 5 hang - 4 cot )');
for i:=1 to n do    (*Chi so hang tu 1-5*)
Begin
for j:=1 to r do    (*Chi so cot tu 1-4*)
Begin
Write('B[',i,',',j,'] = '); read(B[i,j]);
End;
writeln;
End; clrscr;
gotoxy(10,5);
write('DANG NHAN 2 MA TRAN, XIN VUI LONG CHO!' );
Delay(1500);
for i:=1 to m do
for j:=1 to r do
Begin
C[i,j]:=0;
for k:=1 to n do
C[i,j]:=C[i,j]+A[i,k]*B[k,j];
End;
clrscr;
writeln('MA TRAN TICH C = A*B LA ');
For i:= 1 to m do
Begin
for j:= 1 to r do
write(C[i,j]:5); writeln;
End;
repeat until keypressed;
clrscr;
End.
```

BÀI TẬP ỨNG DỤNG CHƯƠNG 5

1. Nhập ví dụ 32 vào máy và cho chạy thử, sau đó sửa chữa chương trình để nhân hai ma trận A(5x7) và B(7x10) . Nếu A và B là các ma trận vuông (ma trận có số hàng và số cột bằng nhau) thì có thể áp dụng chương trình để nhân hay không?

2. Lập chương trình nhập cho mảng M1 50 số thực bất kỳ, sau đó sắp xếp các số đó theo thứ tự tăng dần và giảm dần.

3. Tổng đài điện thoại cơ quan có 4 số. Số đầu luôn bằng 7, số cuối nằm trong khoảng từ 3 đến 7. Hỏi cơ quan đó có thể lắp được bao nhiêu máy? Cho hiện lên màn hình các số máy.

4. Lập chương trình tính tổng bình phương và tổng lập phương của 15 số nguyên đầu tiên.

5. Với 20 tờ giấy bạc mệnh giá 200, 500, 1000, hỏi có bao nhiêu phương án phối hợp các loại tiền để có 13500.

6. Cho mảng a có n phần tử là số thực. Lập chương trình tách mảng a thành bốn mảng theo yêu cầu sau:

   Mảng a1 chứa các số dương

   Mảng a2 chứa các số âm

   Mảng a3 chứa các số chẵn

   Mảng a4 chứa các số lẻ

Chương 6

DỮ LIỆU KIỂU CHUỖI KÝ TỰ (STRING)

I. ĐỊNH NGHĨA CHUỖI KÝ TỰ

Biến kiểu chuỗi ký tự là biến đươc định nghĩa bởi từ khoá STRING kèm theo độ dài tối đa của chuỗi viết trong dấu [...],

Ví dụ

```
Var
Bien1: string[50];          (* Biến1 có độ dài tối đa là 50 ký tự *)
Bien2: string[40];          (* Biến2 có độ dài tối đa là 40 ký tự *)
Bien3: string[100];         (* Biến3 có độ dài tối đa là 100 ký tự *)
```

Có thể xem chuỗi ký tự là một mảng một chiều của các ký tự. Sự khác nhau giữa mảng và chuỗi là ở chỗ khi định nghĩa mảng thì số phần tử của mảng đã được xác định, còn số phần tử chuỗi (tức là số ký tự có trong chuỗi) có thể nhỏ hơn độ dài đã định nghĩa. Một dòng văn bản có thể xem như một mảng của các chuỗi ký tự. Trong tiếng Việt từ dài nhất có 7 ký tự (từ "Nghiêng"), nếu xem như trên một dòng có thể viết nhiều nhất 255 ký tự, giữa các từ có một khoảng trống thì một mảng các từ chỉ có thể chứa 255/8 = 32 từ. Nghĩa là:

```
Dong_van_ban:array[1..32] of string[7];
```

Nếu thực hiện phép gán:

```
Bien1:= 'Cong hoa xa hoi chu nghia viet nam';
```

Thì trong bộ nhớ chuỗi ký tự Bien1 sẽ chiếm 34 Bytes cộng thêm một Byte (Byte số 0) chứa ký tự thể hiện độ dài của chuỗi.

Ví dụ: Byte số 0 trong bộ nhớ chứa một chuỗi nào đó là chữ "a" thì chuỗi có độ dài thực là 97 vì trong bảng mã ASCII chữ a có số thứ tự là 97. Có thể xuy ra ngay một điều là số thứ tự lớn nhất trong bảng mã ASCII là 255 do đó không thể có một chuỗi có độ dài lớn hơn 255 ký tự.

Để kiểm tra độ dài của một chuỗi có thể dùng lệnh:

```
Write(Ord(Bien1[0]));
```

Kết quả sẽ là: 34

Muốn biết ký tự chỉ độ dài của chuỗi là ký tự nào ta có thể dùng hàm CHR(số):

```
Write(chr(Ord(Bien1[0])));
```

Kết quả là nhận được là ký tự " (ký tự " có số thứ tự là 34)

II. CÁC THAO TÁC VỚI CHUỖI KÝ TỰ

1. Truy nhập vào chuỗi

Mỗi phần tử của chuỗi được xác định bởi vị trí của nó trong chuỗi, để truy nhập vào một phần tử nào đó ta phải viết tên biến chuỗi kèm theo vị trí của ký tự đặt trong dấu [..].

Ví dụ để thay thế ký tự v thường thành V hoa trong từ "viet" của chuỗi Bien1 ta có thể viết lệnh

```
if Bien1[27]="v" then Bien1[27]:="V";
```

Số 27 cho biết vị trí của ký tự "v" trong chuỗi Bien1

2. Viết chuỗi ký tự ra màn hình

Để viết một chuỗi ra màn hình có thể dùng một trong hai thủ tục đã biết:

```
Write(Bien1);
```

hoặc:

```
Writeln(Bien1);
```

Nếu sử dụng một biến đếm i kiểu Integer ta cũng có thể viết chuỗi qua vòng lặp xác định FOR:

```
FOR i:=1 to 34 do Write(bien1[i]);
```

3. Cộng chuỗi ký tự

Trở lại ví dụ đã nêu, giả sử rằng ta gán giá trị cho Bien2

```
Bien2:='Doc lap - Tu do - Hanh phuc ';
```

Khi đó ta có thể thực hiện phép cộng hai chuỗi

```
Bien3:=Bien1+Bien2;
```

Lệnh viết hai dòng văn bản Bien1 và Bien2 ra màn hình lúc này có thể tiến hành thông qua việc viết Bien3:

```
FOR i:= 1 TO 34 DO Write(Bien3[i]);
Writeln;
FOR i:=35 TO 100 DO Write(Bien3[i]);
```

Ngoài việc cộng các chuỗi đã khai báo ta còn có thể cộng thêm các từ vào các chuỗi nếu độ dài cho phép, ví dụ:

```
Bien3:=Bien3+'DON XIN VIEC';
```

4. So sánh hai chuỗi ký tự

Hai chuỗi ký tự gọi là bằng nhau nếu các ký tự có cùng vị trí trong hai chuỗi giống nhau và độ dài thực của chúng bằng nhau.

Hai chuỗi "Doc lap" và "Doc lap " là không bằng nhau.

Xét hai chuỗi chuoi1 và chuoi2 trong ví dụ 6.1. Việc so sánh hai chuỗi được tiến hành bắt đầu từ ký tự đầu tiên bên trái , nếu ký tự đầu tiên của chuoi1 lớn hơn ký tự đầu tiên của chuoi2 thì biến BSS cho giá trị FALSE nghĩa là chuoi1>chuoi2, ngược lại thì BSS có giá trị TRUE nghĩa là chuoi1<chuoi2. Nếu ký tự đầu giống nhau thì tiếp tục so sánh ký tự thứ 2, ký tự của chuỗi nào lớn hơn thì chuỗi đó lớn hơn (không cần biết các ký tự sau đó như thế nào.

Lưu ý: *Khoảng trống giữa các từ cũng được xem là một ký tự để so sánh, mã ASCII của phím SPACE (phím khoảng trống) là 32.*

Ví dụ 6.1:

```
Var
BSS: Boolean;     (*BSS Biến so sánh*)
chuoi1: string[30];
chuoi2: string[40];
Begin
chuoi1:= 'Doc lap';
chuoi2:= 'Doc lap tu do';
BSS:= chuoi1<chuoi2;
Writeln(BSS);
chuoi1:= 'To quoc';
BSS:= chuoi1<chuoi2;
Writeln(BSS);
End.
```

Chạy chương trình trên ta có kết quả là biến BSS đầu tiên nhận giá trị TRUE sau đó nhận giá trị FALSE

Trong quá trình so sánh hai chuỗi chỉ số dùng để so sánh là số thứ tự của ký tự trong bảng mã ASCII. Để biết được chỉ số đó có thể tra cứu bảng mã trong các tài liệu Tin học như Hệ điều hành MS-DOS hoặc tự lập chương trình. Dưới đây là một chương trình mà bạn đọc có thể nhập và cho chạy ngay.

Ví dụ 6.2:

```
Program   ma_ascii;
uses crt;
var
x1,x2,x3,i,j:integer;
x:char;
begin j:=1;
x1:=0;
x2:=20; clrscr; x:=' ';
writeln;
repeat
gotoxy(1,j);
for i:=x1 to x2 do write(i:4);
delay(1000);
gotoxy(2,j+1);
```

```
for i:=x1 to x2 do write(x,chr(i),x,x);
delay(1000);
writeln;
x1:=x1+20;
x3:=255-x2;
if x3<20 then x2:=x2+x3
else x2:=x2+20;
j:=j+2;
until x2>=255+x3;   (*repeat until keypressed;*)
 end.
```

Thủ tục DELAY(1000) là thủ tục làm chậm chương trình lại 1000 ms (miligiây) cho ta quan sát. Nếu không dùng DELAY thì phải dùng thủ tục Repeat Until Keypressed (giữ màn hình lại cho đến khi bấm phím bất kỳ).

III. CÁC THỦ TỤC VÀ HÀM XỬ LÝ CHUỖI

Các thủ tục và hàm xử lý chuỗi trong Turbo Pascal 7.0 cũng không khác biệt nhiều so với version 5.0 hoặc 6.0.

1. Phép gán

Phép gán giá trị cho một biến chuỗi cũng dùng toán tử gán ":=", chuỗi ký tự gán cho một biến phải đặt trong dấu nháy đơn ví dụ:

```
Ten:= 'Nguyen Cong Hoan';
Diachi:= ' Khu Hoan kiem - Ha noi ';
```

Nếu chuỗi đem gán có độ dài lớn hơn độ dài đã định nghĩa của biến thì phần đuôi sẽ bị cắt bỏ.

2. Hàm xác định độ dài chuỗi

LENGTH(Tên biến chuỗi):

Hàm này cho ta kết quả là con số chỉ độ dài thực của chuỗi ví dụ:

```
Var
Diachi:String[30];
Begin
Diachi:= ' Khu Hoan kiem - Ha noi ';
LENGTH(DIACHI)
Write(Length(diachi));
end.
```

Kết quả trên màn hình: 24

3. Thủ tục xoá

DELETE(Tên chuỗi, n1, n2); n1 và n2 là các số nguyên

Thủ tục này sẽ xoá đi trong chuỗi đã cho n2 ký tự bắt đầu từ vị trí n1 kéo sang phải, ví dụ:

DELETE(Diachi,15,9);

sau thủ tục này chuỗi Diachi sẽ còn lại là ' Khu Hoan kiem'

4. Thủ tục chèn

INSERT(Chuoi1,Chuoi2,n1);

Thủ tục này sẽ chèn Chuo1 vào Chuoi2 kể từ vị trí n1, ví dụ:

INSERT('Tap the',Diachi,6);

Kết quả biến Diachi sẽ là: Khu Tap the Hoan kiem

5. Chuyển dãy số thành chuỗi ký tự

STR(Quy cách, tên biến chuỗi);

Thủ tục STR(..) sẽ chuyển đổi một số (Real hoặc Integer) thành một chuỗi ký tự biểu diễn chính số đó theo "Quy cách". "Quy cách" ở đây được quy định bao gồm:

- Ten biến số hoặc chính con số cần chuyển
- Dấu hai chấm ":"
- Cách viết có quy cách của các số là cách viết số nguyên hoặc số thực đã nêu trong chương 4.

Tên biến chuỗi là tên chuỗi ký tự chứa các ký tự đã chuyển. Chúng ta có thể tham khảo qua ví dụ 6.3.

Ví dụ 6.3:

```
PROGRAM CHUYENSO;
uses crt;
var   x1:integer;   x2:real;
ten1,ten2,ten3:string[20];
begin
clrscr;
x1:=1234;   x2:=123.456789;
str(x1:15,ten1);
writeln(ten1);
str(x2:7:3,ten2);
writeln(ten2);
str(123456:10,ten3);
writeln(ten3);
repeat until keypressed;
end.
```

Ví dụ 6.3 sẽ viết lên màn hình dòng chữ :

'TRUONG DAI HOC CHU VAN AN *** KHOA CNTT ';

dòng chữ này sẽ chạy trên dòng 10 từ trái qua phải đồng thời đổi màu liên tục theo luật ngẫu nhiên. Chúng ta có thể biến đổi chương trình để dòng chữ chạy từ trên xuống dưới, từ dưới lên trên hoặc hiện lần lượt từng ký tự trên màn hình như ví dụ 6.4.

Ví dụ 6.4:

```
Program chu_chay;
uses crt;
var  b:string[100];   i,j,k:integer;
Begin
b:='TRUONG DAI HOC CHU VAN AN *** KHOA CNTT ';
clrscr;
for i:=1 to 80 do
begin
gotoxy(80-i,10);
textcolor(random(14));
writeln(b);
CLRSCR;
delay(600);
end;
end.
```

BÀI TẬP ỨNG DỤNG CHƯƠNG 6

1. Nhập vào chuỗi S một số ký tự vừa chữ in vừa chữ thường, tách S thành hai chuỗi, một chuỗi chỉ chứa chữ in, một chuỗi chỉ chứa chữ thường.

2. Nhập vào chuỗi S một số ký tự bao gồm cả chữ và số, tách S thành hai chuỗi, một chuỗi chỉ chứa chữ , một chuỗi chỉ chứa số.

3. Cho dòng chữ "CHUC MUNG NAM MOI", lập chương trình hiện từng ký tự trên màn hình tại vị trí ngẫu nhiên với màu ngẫu nhiên sau đó cho dòng chữ chạy từ đỉnh xuống đáy màn hình.

4. S là một chuỗi gồm các chữ cái thường. Viết chương trình sắp xếp các ký tự của chuỗi theo chiều tăng dần từ a đến z

5. Cho hai chuỗi S1 và S2 độ dài của S1 gấp đôi S2. Viết chương trình chèn S1 vào trong S2 sao cho ký tự đầu tiên của S1 nằm ở vị trí giữa của S2

6. Chuỗi S gồm cả chữ cái và chữ số. Viết chương trình đếm xem trong chuỗi có bao nhiêu chữ số, đó là các số nào?

Chương 7

CHƯƠNG TRÌNH CON - THỦ TỤC VÀ HÀM

I. KHÁI NIỆM CHUNG VỀ CHƯƠNG TRÌNH CON

Chương trình con trong Turbo Pascal gồm hai loại: **Thủ tục (Procedure)** và **hàm (Function)**. Các chương trình con được dùng rộng rãi khi xây dựng các chương trình lớn nhằm làm cho chương trình dễ theo dõi, dễ sửa chữa. Mặt khác các chương trình con lại có thể sử dụng chung cho một số chương trình khác, điều này có ý nghĩa to lớn không phải chỉ đối với những người lập trình chuyên nghiệp mà cả đối với người sử dụng.

Bản thân tên gọi của 2 loại chương trình con đã nói lên phần nào sự khác nhau giữa chúng. Function (Hàm) là một loại chương trình con cho kết quả là một giá trị vô hướng. Khi gọi tên Function với các tham số hợp lệ ta sẽ nhận được các giá trị, bởi vậy tên hàm có thể đưa vào các biểu thức tính toán như là các toán hạng. Procedure là một chương trình con khi thực hiện không cho ra kết quả là một giá trị, mỗi Procedure nhằm thực hiện một nhóm công việc nào đó của chương trình chính, vì vậy tên của Procedure không thể đưa vào các biểu thức tính toán. Mỗi chương trình con có thể được soạn thảo riêng và chạy thử trước khi ghép vào thân chương trình chính. Điều này cho phép người lập trình có thể phân mảnh chương trình cho nhiều người cùng làm dưới sự chỉ đạo thống nhất của người xây dựng chương trình chính. Trong Turbo Pascal đã có sẵn một số chương trình con, ví dụ: sin (x), sqrt (x).... là các Function, còn read(), write(), gotoxy (x1,x2)..... là các Procedure.

Trong một chương trình chính các chương trình con được bố trí ngay sau phần khai báo biến theo thứ tự Procedure hay Functiond ta chọn. Cấu trúc tổng quát một chương trình trong Turbo Pascal như sau:

```
PROGRAM tên_chương_trình;
USES tên các UNIT;  (*khai báo các đơn vị chương trình cần dùng*)
LABEL               (*khai báo nhãn*).
CONST               (*Khai báo hằng*)
TYPE                (*định nghĩa kiểu dữ liệu mới*)
VAR                 (*khai báo biến*)
PROCEDURE Tên_CTC1(danh sách tham số hình thức)
 (*CTC thứ nhất khai báo các mục như đối với một chương trình hoàn chỉnh nếu cần*)
Begin
..... (*thân Procedure thứ nhất*).
End;
```

```
PROCEDURE Tên_CTC2(danh sách tham số hình thức)
(*CTC thứ hai*).
..... (*thân Procedure thứ hai*)
FUNCTION Tên_HAM1(danh sách tham số hình thức)
(*Hàm thứ nhất khai báo các mục như đối với một chương trình hoàn chỉnh nếu cần*)
Begin
..... (*thân hàm thứ nhất*)
End;
.....
BEGIN (*bắt đầu chương trình chính*)
.....
END.
```

Ghi chú:

1. Các chương trình con về nguyên tắc cũng bao gồm các phần khai báo báo như đối với một chương trình chính, phần nào không cần thiết thì không khai. Điều khác nhau cơ bản là thân chương trình con nằm giữa hai từ khoá Begin và End; (sau End là dấu ";" chứ không phải là dấu "." như trong chương trình chính) ngoài ra chương trình con còn có thể thêm phần khai báo các tham số hình thức, các tham số hình thức được đặt trong dấu () và viết ngay sau tên chương trình con.

2. Tên các tham số cùng kiểu được đặt cách nhau bởi dấu "," rồi đến kiểu các tham số viết sau dấu ":". Các nhóm tham số khác kiểu viết cách nhau bởi dấu ";". Bạn đọc có thể xem ví dụ 7.1 trong mục II.

II. THAM SỐ TRONG CHƯƠNG TRÌNH CON

Các chương trình con thường sử dụng các giá trị truyền cho chúng qua bàn phím hoặc do chương trình con khác tạo ra, đồng thời nó cũng tạo ra các giá trị cho chương trình con khác sử dụng. Các giá trị này được gọi chung là tham số (**Parameter**). Những tham số mà giá trị của nó có thể biến đổi trong một chương trình con nào đó thì được gọi là tham biến (**Variabic parameter**) trong chương trình con đó. Những tham số mà giá trị của nó không được phép thay đổi trong chương trình con thì được gọi là tham trị (**Value parameter**). Các tham số trong chương trình con được gọi là tham số hình thức, nó sẽ được thay thế bằng các tham số thực trong chương trình chính khi có lời gọi thực hiện chương trình con. Xét câu lệnh sau:

```
PROCEDURE VIDU(x,y,z: integer; lam:boolean; var qq: char);
```

Câu lệnh khai báo chương trình con trên đây đồng thời khai báo các tham số hình thức trong đó x, y,z, lam là các tham trị với x, y,z có kiểu integer. lam có kiểu boolean, qq là tham biến vì nó được viết sau từ khoá VAR.

Ví dụ 7.1: xây dựng chương trình tìm số lớn nhất trong 15 số nguyên được nhập từ bàn phím.

```
PROGRAM TIM_TRI_CUC_DAI;
USES CRT;
```

```
TYPE dayso = array[1..15] of integer;
(*Định nghĩa kiểu dữ liệu dayso là mảng gồm 15  phần tử  nguyên*)
VAR
a: dayso                    (*khai báo biến của chương trình chính*)
n: integer;

PROCEDURE nhapso(m:integer; var x:dayso);
(* Nhập dãy số cần tìm cực đại vào mảng một  chiều x[i]*)
Var i:integer;              (*khai báo biến của chương trình con*)
Begin
writeln('nhap day so gom 15 so kieu integer);
For i:=1 to m Do
(* Giá trị m được truyền từ chương trình qua tham số thực n*).
Begin
write('PHAN TU a [','i,']; realln (x[i]);
End; End;

FUNCTION MAX(m: integer; b:dayso); integer;
 (* Hàm MAX dùng để tìm số lớn nhất trong  dãy số đã nhập, kiểu giá trị của
hàm là kiểu integer *)
VAR
i,t: integer; (* Biến riêng của hàm MAX *)
Begin
t:=b[1];  (* Gán cho biến t giá trị của phần tử thứ nhất của mảng b[i]*)
For i:=2 to m Do
if t<b [i] then t:=b[i];
Max:=t;  (* Gán giá trị cho chính hàm Max*)
End;
BEGIN (* Thân chương trình chính *)
Write('Tong so phan tu cua day so = ');
Readln(n);
NHAPSO(N, A); (* Gọi chương trình con NHAPSO với 2 tham số thực là n và a.
Hai tham số này sẽ thay thế cho hai tham số hình thức m, x trong chương trình con *)
Writeln (' So lon nhat trong day so da nhap = ', MAX(n,a):5);
(* Viết ra giá trị của hàm MAX với 2 tham số thực n,a độ dài số là 5 ký tự *)
Repeat until keypressed;
END.
```

Ví dụ 7.1 là một chương trình bao gồm hai chương trình con, chương trình thứ nhất là một thủ tục (Procedure), chương trình con thứ hai là một hàm (Function).

Chương trình chính có lệnh đọc số phần tử n của mảng dayso (tức là số lượng con số sẽ nhập vào). Vì mảng dãy số gồm nhiều nhất là 15 phần tử nên không thể đọc vào nhiều quá 15 con số.

Sau đó là lệnh gọi chương trình con NHAPSO với 2 tham số thực là n, a, ở đây a là tham biến nghĩa là giá trị của mảng a sẽ được thay đổi trong chương trình con bởi tham số hình thức x[i].

Lệnh viết giá trị lớn nhất của dãy số có kèm lời gọi hàm MAX vì function MAX thực chất trong trường hợp này chỉ là một con số.

Chương trình con nhập vào tham biến x [i] n con số thông qua tham trị m (m=n).

Hàm MAX dùng để tìm con số lớn nất trong các con số đã nhập , lời gọi hàm trong chương trình chính kèm theo việc truyền hai tham số thực là n và a thay thế cho hai tham số hình thức là m và b. Tên hàm được dùng như là một biến trong bản thân hàm khi ta dùng phép gán giá trị MAX:=t;

III. TRUYỀN THAM SỐ CHO CHƯƠNG TRÌNH CON

Trở lại vị trí 7.1 ta thấy trong mỗi chương trình con có những tham số riêng của mình. Chương trình con nhập số đã sử dụng hai tham số m và x. Vì m được khai báo kiểu không có từ khoá Var nên nó là tham trị, nghĩa là suốt quá trình xử lý trong chương trình con giá trị của m có thể thay đổi. "x" là tham biến vì nó được khai báo sau từ khoá Var. Khi tham số hình thức trong chương trình con là tham biến thì tham số thực trong chương trình chính cũng phải là biến, trong trường hợp cả hai tham số thực và tham số hình thức đều phải cùng kiểu dữ liệu.

Các tham số thực truyền cho tham biến thì giá trị của nó có thể thay đổi trong chương trình con, khi ra khỏi chương trình con nó vẫn giữ ngyên các giá trị đã thay đổi đó. Trong ví dụ 7.1 tham số thực a truyền vào tham biến x và bị biến đổi bởi phép gán trong chương trình con NHAPSO, sau khỏi ra chương trình con nó giữ nguyên các giá trị đã gán là các giá trị ta đọc từ bàn phím vào mảng.

Khi tham số hình thức là tham trị thì tham số thực phải là một biểu thức hay cụ thể hơn phải là một giá trị. Chương trình con nhận giá trị này như là giá trị ban đầu và sẽ thực hiện các phép tính làm biến đổi giá trị đó, quá trình này chỉ tác động trong nội bộ chương trình con, khi ra khỏi chương trình con giá trị của tham số thực không biến đổi. Cụ thể trong ví dụ trên biến n nhận giá trị đọc từ bàn phím trong chương trình chính và được truyền cho tham số m trong cả hai chương trình con. Sau lời gọi chương trình con NHAPSO giá trị này có thể bị thay đổi nhưng khi rời NHAPSO đến lời gọi hàm MAX thì n lại vẫn giữ giá trị ban đầu.

Như vậy nếu muốn bảo vệ giá trị một tham số nào đó khi truyền chúng cho chương trình con thì phải qui định chúng là tham trị. Còn nếu muốn nhận lại giá trị mà chương trình con đã sinh ra thay thế cho những giá trị ban đầu của chương trình chính (hoặc là dùng những giá trị của chương trình con thay thế cho biến chưa được gán giá trị trong chương trình chính như ví dụ 7.1) thì tham số phải là tham biến.

Khi xây dựng các chương trình con cần lưu ý một số điểm sau đây:

1. Kiểu của tham số trong chương trình con phải là kiểu đã được định nghĩa sẵn trong Pascal hoặc dã được định nghĩa trong phần đầu của chương trình chính. Trong bản thân chương trình con không thể định nghĩa kiểu dữ liệu mới.

2. Chương trình con có cần tham số hay không ? Nếu chương trình con chỉ sử dụng các biến đã được khai báo trong chương trình chính (biến toàn cục) thì không cần tham số. Nếu chương trình con thực hiện nhiều công việc trên cùng một loại đối tượng (nghĩa là lời gọi chương trình con được lặp lại nhiều lần trên cùng một nhóm biến) thì cần khai báo tham số hình thức.

3. Nếu không muốn thay đổi giá trị của các tham số thực trong chương trình chính khi truyền nó cho chương trình con thì tham số hình thức trong chương trình con phải là tham trị (trong phần khai báo kiểu không có từ khoá Var). Nếu cần thay đổi giá trị của tham số trong chương trình chính và nhận lại giá trị mà chương trình con đã xử lý thì tham số trong chương trình con phải là tham biến (tên tham số phải đặt sau từ khoá Var.

IV. BIẾN TOÀN CỤC VÀ BIẾN CỤC BỘ

1. Biến toàn cục

Biến khai báo ở đầu chương trình chính được gọi là biến toàn cục. Nó có tác dụng trong toàn bộ chương trình, kể cả các chương trình con. Khi thực hiện chương trình máy dành một ô nhớ để lưu giữ giá trị của biến. Trong ví dụ 38 biến a và n là biến toàn cục, hai biến này có thể sử dụng trực tiếp trong thủ tục NHAPSO và trong hàm MAX mà không cần khai báo lại.

2. Biến cục bộ

Những biến khai báo trong chương trình con chỉ có tác dụng trong nội bộ chương trình con đó, chúng được gọi là biến cục bộ. Biến cục bộ có thể trùng tên với biến toàn cục song máy dành một ô nhớ khác để lưu giữ các giá trị của biến. Khi kết thúc chương trình con máy thu hồi ô nhớ của biến cục bộ dùng vào việc khác ta truy nhập vào biến cục bộ thì máy sẽ báo lỗi unknown indentifict - biến không xác định . Trường hợp trong chương trình con lại có chứa các chương trình con khác thì biến cục bộ của chương trình con mẹ lại được xem là biến toàn cục đối với chương trình con của nó.

Một biến sau khi được khai báo trong một chương trình sẽ chỉ có tầm tác dụng trong bản thân chương trình đó và các chương trình con của nó. Biến này không có tác dụng trong các chương trình cùng cấp khác hoặc trong các chương trình con của chương trình khác.

Khi kết thúc một chương trình con, nếu tham số là tham trị thì ô nhớ chứa giá trị của biến được giải phóng, còn nếu tham số là tham biến thì giá trị của nó sẽ được chép sang ô nhớ của biến toàn cục tương ứng.

V. TÍNH ĐỆ QUI CỦA CHƯƠNG TRÌNH CON

Trong một chương trình con có thể có lời gọi tới chính tên chương trình con đó. Tính chất này được gọi là tính "Đệ qui của chương trình con". Ví dụ kinh điển để nghiên cứu tính đệ quy là phép tính giai thừa.

5! = 1*2*3*4*5

Tổng quát n! = 1*2*3*.....*(n-2)*(n-1)*n

Đặc biệt nếu n = 0 thì 0! = 1

Có thể dễ dàng suy ra rằng:

n! = (n-1)!*n từ tính chất trên ta lập chương trình để tính n!

Ví dụ 7.2:

```
PROGRAM GIAITHUA;
Uses crt;
Var n: integer;
```

```
FUNCTION GT(M: integr): integer; (* Hàm GT tính giai thừa của n*)
Begin
if m=0 then GT:=1 else GT:= m*GT(m-1); (*Gọi đệ qui của GT *)
End;

Begin
clrscr;
write ('TINH GIAI THUA VOI n = '); Readln (n) (*Dọc giá trị n*)
write('Giai thua cua ',n,' = 'GT(n)); (* Viết giá trị hàm GT *)
repeat until keypressed;
End.
```

Ví dụ 7.2 có một số điều đáng lưu ý sau đây:

1. Hàm GT được xây dựng để tính giai thừa với tham số hình thức m, kiểu của m là kiểu Integer. Giá trị m sau này sẽ được thay thế bằng tham số thực n qua lời gọi GT(n) trong chương trình chính.

2. Khi định nghĩa kiểu của GT là Integer thì n chỉ được chọn nhỏ hơn 8 vì 8! = 40320 vượt quá giá trị mà Pascal có thể xử lý (32767). Để có thể tính giai thừa với n>=8 ta phải định nghĩa function GT có kiểu Real, khi đó lệnh viết giá trị của giai thừa phải là viết số thực với phần lẻ bằng 0, ví dụ:

```
Write (GT(n):12:0);
```

3. Việc tính toán có thể sử dụng phép đệ quy được thực hiện như sau:

Khi có lệnh gọi kiểu đệ quy máy sẽ sử dụng bộ nhớ kiểu LIFO (Last In - First Out) đó là bộ nhớ kiểu xếp chồng. Để dễ hình dung ta xét việc tính 3!

FUNCTION GT (n) với n = 3 có lệnh gán GT(3): = 3*GT(2), đến đây máy sẽ ghi vào LIFO các dữ liệu có liên quan đến việc tính GT(3) sau đó tiếp tục tính GT(2). Việc tính GT(2) lại liện quan đến tính GT(1) theo công thức đệ quy đã nêu, máy sẽ ghi tiếp các dữ liệu phục vụ việc tính GT (2) vào LIFO và chuyển sang tính GT (1). Theo công thức đệ quy GT (1): = 1*GT(0), giá trị GT (0) = 1 theo định nghĩa giai thừa, do đó GT(1) = 1. Đến đây diễn ra quá trình tính ngược từ Gt(1) máy tính ra GT(2), từ GT(2) tính ra GT(3).

4. Quá trình tính toán theo thuật toán đệ quy đòi hỏi người lập trình phải hiểu cặn kẽ vấn đề và phải có một tư duy logic sáng sủa, vì vậy nếu sử dụng các vòng lặp có thể giải được bài toán thì nên dùng vòng lặp. Trừ trường hợp bắt buộc phải giải bài toán không có tính lặp hoặc bài toán có khả năng truy hồi.

Ví dụ 7.3 : Lập chương trình tìm ước số chung lớn nhất của hai số nguyên n, m. Ước số chung lớn nhất của hai số n và m tính theo công thức :

$$USC(n,m) = \begin{cases} n \text{ nếu } m = 0 \\ USC(m, \text{phần dư của } n/m) \text{ nếu } m <> 0 \end{cases}$$

Ví dụ n= 4, m=8

USC(4,8) = USC(8, 4) = USC(4,0) = 4

Chương trình được xây dựng như sau:

```
Program timUSC ;
Uses crt;
Var
 n,m : word;
 Lam: Char;
  FUNCTION USC(a,b:word): word;
  Begin
  If  b=0 then USC := a Else USC := USC(b,a mod b);
  End;
BEGIN
  Repeat
  clrscr;
   Textcolor(5); Textbackground(White);
   Write('  Hay cho hai so "n" va "m" can tim uoc so chung ');
   Readln(n,m);
   Writeln('  Uoc so chung lon nhat cua hai so ',n,' va ',m, ' la ',USC(n,m):5);
   Writeln;
   Write('Tim tiep hay thoi ? C/K '); read(lam);
  Until Upcase(lam)='K';
 END.
```

VI. ĐƠN VỊ CHƯƠNG TRÌNH (UNIT)

Khi xây dựng một chương trình lớn chúng ta có thể nhóm một hoặc một số chương trình con cùng với các khai báo kiểu, biến, dử liệu thành một đơn vị chương trình. Các đơn vị chương trình được lưu trữ trong tệp riêng và được gọi ra sử dụng bởi lệnh uses. Nếu chúng ta sử dụng trình biến dịch COMPILE để dịch các chương trình nguồn sang dạng mã máy thì chương trình chính sẽ có đuôi là .exe còn đơn vị chương trình sẽ có đuôi là .TPU. Turbo Pascal 7.0 đã thiết kế sẵn một số đơn vị chương trình thông dụng như: SYSTEM, GRAPH, DOS, CRT, PRINTER, OVERLAY, MEMORY, VIEWS, DIALOGS, DRIVERS,..... các đơn vị này được gọi là đơn vị chuẩn và được lưu trữ trong tệp TURBO.TPL . Ngoài ra Turbo Pascal 7.0 còn một số đơn vị chương trình độc lập để trong thư mục như GRAPH.TPU, BOLD.CHR, EGAVGA.BGI....

Để tìm hiểu một đơn vị chuẩn chứa những thủ tục gì hoặc làm công việc gì có thể sử dụng một phần mềm soạn thảo văn bản bất kỳ (Bked, nc....) truy nhập vào thư mục con DOS trong thư mục chương trình Turbo Pascal 7.0, kế đó gọi các tệp có tên tương ứng (đuôi tệp là .INT) ra màn hình để xem. Mỗi chương trình con chứa trong các đơn vị chuẩn đều có lời giải thích kèm theo kiểu các tham số phải khai báo để người lập trình tiện tham khảo.

Dưới đây giới thiệu sơ lược về các đơn vị chuẩn của Pascal để người đọc nghiên cứu:

1. SYSTEM: Đơn vị SYSTEM là đơn vị cơ bản của Turbo Pascal trong đó cài đặt toàn bộ các thủ tục và hàm mà chúng ta đã sử dụng từ đầu chương trình, system đsx được thiết kế để chạy tự động khi có một chương trình được thực hiện do đó không cần lệnh goi USES ở đầu chương trình.

Trong UNIT SYSTEM có một số thủ tục hay dùng khi lập trình như sau:

1.1. Thủ tục EXIT: Khi gặp EXIT trong một chương trình thì kết thúc chương trình và ra khỏi chương trình đó, nếu đặt exit trong chương trình chính thì dừng toàn bộ công việc.

1.2. Thủ tục CHDIR (tên thư mục): thủ tục CHDIR (change Directory) cho phép chuyên thư mục có tên viết trong dấu ngoặc đơn thành thư mục hiện thời.

1.3.Thủ tục RMDIR (tên thư mục): thủ tục RMDIR (remove Directory) cho phép xoá một thư mục rỗng trên ổ đĩa hiện thời.

1.4. Thủ tục Reneme (<tên cũ>; <Tên mới>): Thủ tục này đổi tên tệp ngoại vi từ tên cũ thành tên mới.

2. DOS : DOS là UNIT chứa các thủ tục liên quan đến hệ điều hành và quản lý tệp:

2.1. SETDATE(năm, tháng, ngày), thủ tục này sẽ đặt lại ngày tháng năm cho đồng hồ của máy, ví dụ: SETDATE (1998, 11, 25);

Lưu ý rằng chỉ có thể đặt lại cho các năm trong phạm vi từ năm 1980 - 2099 nếu đưa vào giá trị khác máy sẽ báo lỗi.

2.2. SETTIME (giờ, phút, giay, phần trăm giây): đặt lại giờ cho đồng hồ của máy.

2.3. GETTIME (giờ, phút, giây, phần trăm giây): cho biết giờ hệ thống, muốn viết ra giờ hệ thống phải dùng một thử nghiệm sau

```
Var x1, x2, x3, x4: wrod;
begin
Gettime (x1, x2, x3, x4);
Write (x1:3, x2:3, x3:3, x4:3);
end.
```

3. CRT: Đơn vị CRT là đơn vị liên quan đến các thủ tục trình bày màn hình mà ta đã biết, CRT không được nạp tự động vì vậy ta phải gọi ra bằng lệnh USES CRT; trong CRT ta có thể sử dụng các thủ tục:

3.1. Gotoxy(m, n): chuyeenr con trỏ tới toạ độ cột m và hàng n. Cần lưu ý rằng 0<=m<=80; 0<=n<=25.

3.2. CLRSCR: xoá sạch màn hình đưa con trỏ về toạ độ (1,1)

3.3. CLREOL: xoá các ký tự từ vị trí hiện thời đến hết dòng

3.4. SOUND (n): Tạo ra một âm thanh với tần số n Hz.

3.5. DELAY (n): Làm chậm (hoặc kéo dài) lệnh phía trước n mili giây (1 giây = 1000 mili giây). Giá trị lớn nhất cho phép là 65 535 ms (tức là giá trị số nguyên lớn nhất mà máy có thể xử lý)

3.6. NOSOUND: Tắt tiếng kêu do SOUND (n) tạo ra

Ba thủ tục sound, delay, nosound được sử dụng để tạo ra các bản nhạc, cần lưu ý rằng sau thủ tục sound và delay âm thanh được tạo ra sẽ kéo dài liên tục nếu không có nosound mặc dù trong delay ta đã chỉ định khoảng thời gian kéo dài là n ms, chỉ khi có thủ tục nosound thì âm thanh mới kéo dài đúng n, ms.

3.7. LOWVIDEO (làm tối màn hình), HIGHVIDEO (làm sáng thêm) NORMVIDEO (trả về trạng thái bình thường).

4. PRINTER : Đơn vị chuẩn Printer có một phần khai báo máy in phục vụ việc in ấn dữ liệu ra giấy, máy in được cài đặt tên là LST.

Khi cần in ấn ta phải có lời gọi USES PRINTER; sau đó trong các lệnh Write hoặc writeln phải đưa tên máy in vào trước các tên biến cần in, ví dụ:

```
Writeln (lst, 'Ket qua tinh toan:', x1);
```

Dòng lệnh trên sẽ in ra giấy dòng chữ: "Ket qua tinh toan:" và giá trị biến của x1.

BÀI TẬP ỨNG DỤNG CHƯƠNG 7

1. Xây dựng hàm tính luỹ thừa n của số thực a, viết kết quả với 4 số lẻ.
2. Xây dựng hàm tính Tg (α) và Cotg (α) trong đó góc α tính theo độ.
3. Lập chương trình tính diện tích các hình tròn, chữ nhật, tam giác trong đó diện tích mỗi hình tính bằng 1 chương trình con.
4. Lập chương trình giải phương trình bậc 2 có sử dụng các chương trình con.
5. Lập chương trình tính căn bậc n của một số dương.
6. Lập chương trình tính tiền gửi ngân hàng, nhập vào bộ nhớ các giá trị v (vốn), l (lãi suất), t (số tháng) . Tính số tiền có hàng tháng và lãi sau t tháng. Phần nhập dữ liệu nằm trong chương trình mẹ, phần tính toán nằm trong chương trình con.
7. Cho 3 điểm $A(x_1,y_1)$, $B(x_2,y_2)$, $C(x_3,y_3)$ không thẳng hàng trên màn hình. Lập chương trình xây dựng các hàm ca, cb, cb tính khoảng cách giữa các điểm và thủ tục tính diện tích, chu vi tam giác ABC.
8. Lập chương trình với ba thủ tục: Nhap, Noi, Sapxep. Thủ tục nhập dùng để nhập vào hai mảng A, B mỗi mảng không quá 100 số thực. Thủ tục Noi dùng để nối A với B thành mảng C. Thủ tục Sapxep sẽ sắp các mảng theo chiều tăng dần. Viết các mảng A, B ban đầu và các mảng đã sắp xếp ra màn hình.
9. Lập chương trình tính:

   X = 1! + 2! + ... + n! với n nhập từ bàn phím.

   Chương trình bao gồm:

   Hàm Max tính giá trị tối đa của n,

   Thủ tục Kiemtra dùng để kiểm tra giá trị n nhập từ bàn phím, nếu n > Max thì yêu cầu nhập lại giá trị n.

   Hàm Tinh dùng để tính X.

   Thông báo kết quả tính ra màn hình.

Chương 8

DỮ LIỆU KIỂU BẢN GHI (RECORD)

I. KHÁI NIỆM CHUNG

Bản ghi là một cấu trúc bao gồm một số (cố định hoặc thay đổi) các phần tử có kiểu khác nhau nhưng liên quan với nhau. Các phần tử này gọi là các trường. Ví dụ bảng điểm của lớp học bao gồm các trường Hoten, Gioitinh, Lop, Diachi, Toan, Ly, Hoa,...., các trường này tạo nên dữ liệu cho một người trong đơn vị và được gọi là một bản ghi. Kiểu dữ liệu của các trường trong Record có thể hoàn toàn khác nhau và được khai báo sau tên trường, những trường có cùng kiểu dữ liệu có thể khai báo cùng trên một dòng phân cách bởi dấu phảy ",". Cuối mỗi khai báo trường phải có dấu ";"

II. KHAI BÁO

Kiểu dữ liệu Record được khai báo ở phần TYPE bao gồm:

<Tên kiểu > = RECORD
<Tên trường 1>: Kiểu;
<Tên trường 2>: Kiểu;
.....
<Tên trường n>: Kiểu;
END;

Ví dụ 8.1:

```
TYPE
BANGDIEM = RECORD
Hoten: String[25];
Gioitinh: Char;
Lop: String[5];  Diachi: String[50];
Toan,Ly,Hoa: Real;
END;
```

III. TRUY NHẬP

Sau khi đã khai báo kiểu dữ liệu (như trong ví dụ 8.1 BANGDIEM là dữ liệu kiểu bản ghi) ta phải khai báo biến, giả sử cần quản lý danh sách một lớp học nghĩa là ta cần một biến Danhsach viết tắt là DS. Khi đó ta phải khai

```
VAR
DS: Bangdiem;
.....
```

Mặc dù DS là một biến nhưng chúng ta không xử lý chính biến đó mà chỉ xử lý các trường của biến DS. Để truy nhập vào trường cần viết:

<Tên biến>.<Tên trường>

Ví dụ để nhập dữ liệu cho trường Hoten ta viết các lệnh:

Write(' Ho va ten hoc sinh: '); Readln(DS.hoten);

Lệnh Readln(DS.hoten); cho phép ta gán Họ tên học sinh vào trường Hoten của bản ghi đầu tiên. Nếu trên máy có các chương trình mã bàn phím tiếng Việt thường trú ta có thể viết họ và tên bằng tiếng Việt.

Với các trường khác khi truy nhập ta cũng vẫn phải ghi tên biến Record trước tên trường điều này làm mất thêm thời gian và rườm rà chương trình do đó người ta khắc phục nó bằng cách đưa thêm vào lệnh WITH... DO.

IV. LỆNH WITH...DO

Cú pháp của lệnh:

WITH <Tên biến kiểu RECORD> DO <Các lệnh>

Khi xử dụng lệnh WITH...DO chuỗi lệnh viết sau DO chỉ cần viết tên trường có liên quan. Xét ví dụ nhập điểm cho lớp học với giả thiết là lớp có 40 học sinh.

Ví dụ 8.2:

```
PROGRAM NHAPDIEM;
Uses CRT;
Type
BANGDIEM = RECORD
Hoten: String[25];
Gioitinh: Char;
Lop: String[5];
Diachi: String[50];
Toan,Ly,Hoa: Real;
END;
Var
DS_LOP: Array[1..40] of  BANGDIEM; (*danh sách lớp là mảng 40 phần tử*)
i,j: Integer;  lam: Char;
Begin
clrscr;
lam:='C';
i:=1;
Repeat
With DS_LOP[i] DO
Begin
```

```
Write(' Ho va ten hoc sinh: '); Readln(Hoten);
Write(' Trai hay gai T/G: '); Readln(Gioitinh);
Write(' Thuoc lop: '); Readln(Lop);
Write(' Cho o thuong tru: '); Readln(Diachi);
Write(' Diem toan: '); Readln(Toan);
Write(' Diem ly: '); Readln(Ly);
Write(' Diem hoa: '); Readln(Hoa);
End;
i:=i+1;
Write(' NHAP TIEP HAY THOI ? C/K '); Readln(lam);
Until upcase(lam)='K';
clrscr;
For j:=1 to i-1 do
With DS_LOP[j] DO
Writeln(Hoten:15,' ',Gioitinh:2,' ',Lop:4,' ',Diachi:10,' Toan:', Toan:4:2,' Ly:',Ly:4:2,' Hoa:',Hoa:4:2);
REPEAT UNTIL KEYPRESSED;
End.
```

Còn hai kiểu bản ghi chưa đề cập trong giáo trình này là bản ghi lồng nhau và bản ghi có cấu trúc thay đổi. Nếu độc giả muốn nghiên cứu xin mời tìm hiểu trong giáo trình Lập trình nâng cao của cùng tác giả do Nhà xuất bản Nông nghiệp phát hành năm 2005.

BÀI TẬP ỨNG DỤNG CHƯƠNG 8

1. Nhập vào thư viện n đầu sách (n<=50), các thông số cần đưa vào là: Tên sách, Tên tác giả, Năm xuất bản, Tên nhà xuất bản, Số trang, Giá tiền. Hiện danh mục sách đã nhập lên màn hình.
2. Dữ liệu tuyển sinh gồm các trường: SBD, HOTEN, TOAN, LY, HOA, TONG, KETQUA.

   Lập chương trình nhập dữ liệu cho một phòng thi không quá 40 thí sinh, số liệu trong trường Tong = Toan+Ly+Hoa, trường Ketqua điền vào chữ "DO" nếu tổng >=20, còn lại là "TRUOT".

Hiện lên màn hình những người DO (đỗ)

3. Dữ liệu quản lý hàng hoá bao gồm: Mã hàng, Tên hàng, Số lượng, Đơn giá, Ngày nhập, Số chứng từ.

Lập chương trình nhập hàng vào kho, sau mỗi mặt hàng nhập lại hỏi: "Nhap tiep hay thoi? C/K ". Nếu bấm "K" thì kết thúc nhập và hiện danh mục hàng đã nhập lên màn hình.

4. TOADO là kiểu bản ghi chứa toạ độ (cột,dòng) của ba điểm trên màn hình. Lập chương trình kiểm tra xem ba điểm này có thẳng hàng không, nếu không thẳng hàng thì tạo nên tam giác gì (Vuông, Cân, ...)
5. Dữ liệu quản lý vé xe lửa gồm: Số toa, số ghế, giá vé, ga đi, ga đến, Đã bán vé chưa. Mỗi toa tầu có 50 ghế đánh số thứ tự từ 1 đến 50.

Lập chương trình nhập vào những ghế đã bán vé, thông báo còn bao nhiêu chỗ chưa bán vé.

6. Dữ liệu quản lý hàng hoá bao gồm: Mã hàng, Tên hàng, Số lượng, Đơn giá, Ngày nhập, Số chứng từ. Ngày nhập bao gồm Ngay, Thang, Nam.

Lập chương trình nhập hàng vào kho, kết thúc nhập khi mã hàng bằng dấu "*" . Cho một tháng nào đó, hiện những hàng đã nhập trong tháng đó.

7. Dữ liệu quản lý đất đai bao gồm hai trường Chủ đất và Đất. Chủ đất bao gồm: Mã hồ sơ, Họ tên, Mã đất, Địa chỉ. Đất bao gồm: Mã đất, Diện tích, Loại đất. Nhập dữ liệu cho n người (n<=30) sau đó hiện lên màn hình những người sử dụng nhiều đất nhất.
8. Bảng kê hàng hóa bán ra bao gồm các cột: Mã hàng, tên hàng, số lượng bán, đơn giá, thành tiền. Tiền bán = số lượng bán * đơn giá. Viết chương trình nhập n mặt hàng đã bán. Tính tiền bán và hiện lên màn hình.

Chương 9

DỮ LIỆU KIỂU TỆP (FILE)

I. KHÁI NIỆM VỀ TỆP

Tệp dữ liệu là một dãy các phần tử cùng kiểu được sắp xếp một cách tuần tự. Tệp dữ liệu được cất giữ ở bộ nhớ ngoài (đĩa mềm hoặc đĩa cứng) dưới một tên nào đó, cách đặt tên tuân theo quy định của DOS nghĩa là phần tên tệp dài không quá 8 ký tự và phần đuôi không quá ba ký tự.

Tệp tập hợp trong nó một số phần tử dữ liệu có cùng cấu trúc giống như mảng (Array) song khác mảng là số phần tử của tệp chưa được xác định.

Trong Pascal có 3 loại tệp được sử dụng là:

1- Tệp có kiểu

Tệp có kiểu là tệp mà các phần tử của nó có cùng độ dài và cùng kiểu dữ liệu. Với những tệp có kiểu chúng ta có thể cùng một lúc đọc dữ liệu từ tệp ra hoặc nhập dữ liệu vào tệp.

2- Tệp văn bản (Text)

Tệp văn bản dùng để lưu trữ dữ liệu dưới dạng các ký tự của bảng mã ASCII, các ký tự này được lưu thành từng dòng, độ dài của các dòng có thể khác nhau. Khi ghi một số nguyên, ví dụ số 2003 (kiểu word) vào tệp văn bản, Pascal cần 4 byte cho bốn ký tự chứ không phải là 2 byte cho một số. Việc ghi các số nguyên hoặc thực vào tệp văn bản sẽ phải qua một công đoạn chuyển đổi, điều này sẽ do Pascal tự động thực hiện. Để phân biệt các dòng Pascal dùng hai ký tự điều khiển là CR - về đầu dòng và LF - xuống dòng mới.

Trong bảng mã ASCII ký tự CR = CHR(13) còn LF = CHR(10)

3- Tệp không kiểu

Tệp không kiểu là một loại tệp không cần quan tâm đến kiểu dữ liệu ghi trên tệp. Dữ liệu ghi vào tệp không cần chuyển đổi.

Khi khai báo một biến kiểu tệp thì đồng thời ta cũng phải tạo ra một mối liên hệ giữa biến này và một tệp dữ liệu lưu trên thiết bị nhớ ngoài. Cách khai báo này đã được chuẩn hoá trong Pascal.

Kiểu dữ liệu của các phần tử trong biến kiểu tệp có thể là bất kỳ kiểu nào trừ kiểu của chính nó tức là trừ kiểu tệp. Ví dụ nếu kiểu phần tử là một mảng một chiều của các số nguyên ta có tệp các số nguyên, nếu kiểu phần tử là Record ta có tệp các Record...

Tác dụng lớn nhất của kiểu dữ liệu tệp là ta có thể lưu trữ các dữ liệu nhập vào từ bàn phím cùng các kết quả xử lý trong bộ nhớ RAM ra tệp để dùng nhiều lần. Các kiểu dữ liệu đã học chỉ xử lý trong RAM và in kết quả ra màn hình hoặc ra máy in, khi kết

thúc chương trình hoặc mất điện cả dữ liệu nhập vào và kết quả xử lý đều bị mất. Do hạn chế của chương trình chúng ta sẽ chỉ nghiên cứu hai loại tệp là tệp văn bản và tệp có kiểu

II. KHAI BÁO

Tệp có kiểu được định nghĩa sau từ khoá TYPE, còn biến kiểu tệp được khai báo sau từ khoá VAR. Thủ tục khai báo biến kiểu tệp gồm hai cách:

1. Định nghĩa kiểu tệp với từ khoá FILE OF trong phần mô tả kiểu sau từ TYPE, tiếp theo là khai báo biến tệp trong phần khai báo biến

Ví dụ 9.1

```
Type
MSN = Array[1..100] of integer;      (*định nghĩa mảng số nguyên*)
TSN = File of MSN; {định nghĩa tệp TSN có các phần tử là mảng số nguyên}
CHUVIET = File of String[80];
  (* định nghĩa CHUVIET là tệp các chuỗi có độ dài 80 ký tự *)
BANGDIEM = Record
   Hoten: String[25];
   Gioitinh: Char;
   Lop: String[5];
   Diachi: String[50];
   Toan,Ly,Hoa: Real;
END;
TBD = File of BANGDIEM;
VAR
Tep1: TSN;          (* biến tep1 có các phần tử là mảng số nguyên *)
Tep2: CHUVIET;      (* biến Tep2 có các phần tử là chuỗi ký tự *)
Tep3: TBD;          (* biến Tep3 có các phần tử là record*)
.....
```

2. Định nghĩa trực tiếp biến kiểu tệp trong phần khai báo biến

Ví dụ:

```
Var
Tep4: File of Array[1..5] of String[80]; Tep5: File of BANGDIEM;
```

Trong phần khai báo này biến Tep4 là một biến kiểu tệp, tệp này có 5 phần tử, mỗi phần tử là một chuỗi dài tối đa 80 ký tự. Biến Tep5 là một biến tệp mà các phần tử của nó có kiểu Bangdiem.

III. TRUY NHẬP VÀO TỆP

Các phần tử của tệp được lưu giữ tuần tự thành một dãy và việc truy nhập vào từng phần tử phụ thuộc vào thiết bị ghi, đọc của máy vi tính. Turbo Pascal có thể xử lý hai loại tệp là: **Tệp truy nhập tuần tự và tệp truy nhập trực tiếp.**

* **Tệp truy nhập tuần tự**: để truy nhập vào một phần tử nào đó ta bắt buộc phải đi qua các phần tử trước đó. Nếu muốn thêm các phần tử vào tệp thì chỉ có thể thêm vào cuối tệp. Tệp kiểu này dễ hình dung, dễ sử dụng song không linh hoạt, tốn thời gian xử lý. Việc truy nhập tuần tự thường được thực hiện thông qua một vòng lặp. Tệp văn bản là tệp thuộc kiểu này.

* **Tệp truy nhập trực tiếp**: là tệp có thể truy nhập vào phần tử bất kỳ trong tệp, trên những thiết bị nhớ ngoài cổ điển như băng từ, băng đục lỗ không thể tạo tệp kiểu này vì không thể đọc ngay vào giữa băng. Chỉ những máy sử dụng đĩa (mềm hoặc cứng) thì mới có thể tạo tệp truy nhập trực tiếp vì có thể điều chỉnh để đầu từ đặt đúng vào một cung từ chứa dữ liệu nào đó. Muốn truy nhập trực tiếp phải dùng thủ tục Seek(số hiệu phần tử).

IV. MỞ TỆP

Để mở một tệp chuẩn bị lưu trữ dữ liệu Pascal sử dụng hai thủ tục chuẩn sau đây:

ASSIGN(biến tệp, tên tệp);

*(*liên kết biến tệp với một tên tệp sẽ ghi vào thiết bị nhớ ngoài *)*

REWRITE(biến tệp);

(tạo một biến tệp rỗng chuẩn bị nhập dữ liệu vào *)*

Trong đó:

* Biến tệp: là tên biến tệp đã khai báo sau từ khoá VAR

* Tên tệp: Là tên do ta chọn để ghi dữ liệu vào đĩa (theo quy định của DOS). Tên tệp có thể bao gồm cả đường dẫn tới thư mục mà chúng ta lựa chọn. Đường dẫn và tên tệp phải đặt trong dấu nháy đơn. Ví dụ:

ASSIGN(f,'a:\baitap.txt');

Sau thủ tục REWRITE ta sẽ có một tệp rỗng vì chưa có phần tử nào được đọc vào. Nếu trên thiết bị nhớ ngoài ta đã có sẵn một tệp trùng tên với tên ghi ở thủ tục ASSIGN thì tệp ngoài sẽ bị xoá đi.

Sau hai thủ tục chuẩn bị trên đây, để tiến hành ghi dữ liệu vào tệp ta lại dùng thủ tục WRITE(....) như đã biết.

Cách viết:

WRITE(biến tệp, các giá trị cần ghi vào tệp);

Bước cuối cùng là phải đóng tệp lại bằng thủ tục

CLOSE(biến tệp);

Sau thủ tục này dữ liệu sẽ được lưu trên đĩa và tệp sẽ bị đóng.

V. TỆP VĂN BẢN

1. Khai báo tệp văn bản

Tệp văn bản được khai báo trực tiếp trong phần khai báo biến:

Var

Bien_tep : Text;

Ví dụ:

```
Var
Hoso: Text;
T1,T2,T3:Text;
```

2. Truy nhập vào tệp

Truy nhập vào tệp được hiểu là nhập dữ liệu vào tệp, ghi lại dữ liệu trên thiết bị nhớ ngoài, đọc dữ liệu đã có ra màn hình hoặc máy in và xử lý dữ liệu đó.

Đối với tệp văn bản việc ghi dữ liệu vào tệp có thể thực hiện qua hai cặp thủ tục sau:

2.1. Mở tệp mới để ghi

```
Assign(bien_tep, Đường dẫn\ten_tep);
Rewrite(bien_tep);
```

Cặp thủ tục trên dùng để mở một tệp mới chuẩn bị nhận dữ liệu, nếu trong bộ nhớ ngoài đã có tệp trùng tên thì tệp ở ngoài sẽ bị xoá. Nếu bỏ qua *đường dẫn* thì tệp sẽ được lưu vào thư mục hiện hành tức là thư mục mà từ đó Pascal đã được khởi động (C:\TP\BIN). Giả sử chúng ta muốn lưu tệp với tên là BT1.DAT vào thư mục BAITAP trên đĩa A thì phải viết lệnh Assign(bien_tep, 'A:\BAITAP\BT1.DAT');

2.2. Mở tệp đã có để ghi thêm

```
Assign(bien_tep, ten_tep);
Append(Bien_tep);
```

Cặp thủ tục trên mở một tệp đã lưu trong thiết bị nhớ ngoài để nhập thêm dữ liệu, nếu tệp không có thì máy sẽ thông báo lỗi. Dữ liệu nhập thêm sẽ được ghi vào cuối tệp.

2.3. Mở tệp để đọc dữ liệu

Dưới đây là cặp thủ tục dùng để mở tệp đã tồn tại, chuẩn bị cho việc đọc dữ liệu từ tệp ra thiết bị ngoài:

```
Assign(bien_tep, ten_tep);
Reset(bien_tep);
```

3. Ghi dữ liệu vào tệp

Sau khi đã mở tệp chúng ta có thể dùng thủ tục Write hoặc Writeln để ghi dữ liệu vào tệp.

Cú pháp:

```
Write(Bien_tep, giá trị 1, giá trị 2, ...);
```

Hoặc

```
Writeln(Bien_tep, giá trị 1, giá trị 2, ...);
```

Sự khác nhau giữa Write và Writeln là ở chỗ thủ tục Write sẽ ghi dữ liệu liên tục trên các ô nhớ, không có các ký hiệu về đầu dòng và xuống dòng. Thủ tục Writeln sẽ đưa thêm vào cuối dòng các ký tự điều khiển CR (về đầu dòng) và LF (nhảy xuống dòng dưới).

Giá trị 1, giá trị 2... trong lệnh ghi có thể là chuỗi ký tự viết tường minh, tên biến hoặc biểu thức, các giá trị phải được phân cách ít nhất bằng một khoảng trống.

Thủ tục Writeln(Bien_tep); sẽ tạo nên một dòng rỗng không chứa ký tự nào cả.

Dữ liệu trước khi ghi vào tệp văn bản có thể thuộc các kiểu đơn giản sau đây: ***Số nguyên, số thực, ký tự, chuỗi, logic.*** Dù thuộc kiểu gì thì khi đã ghi vào tệp đều được chuyển đổi thành kiểu ký tự. Điều này cũng đồng nghĩa với quy định rằng các kiểu dữ liệu có cấu trúc như: ***Array, Set, Record, File*** không thể lưu trữ trên tệp văn bản.

Khi có thủ tục Close(bien_tep) Pascal sẽ đưa thêm vào dòng cuối cùng ký hiệu kết thúc tệp EOF (End of File).

Ví dụ 9.2

```
Var
T1:text;
Begin
Assign(T1,'DULIEU.DAT');
Rewrite(T1);
Writeln(t1,'Tep van ban');
Write(T1,123);
Write(T1,' ',123.45);
Writeln(T1);
Close(T1);
End.
```

Ví dụ 9.2 tạo ra tệp văn bản DULIEU.DAT tệp này sẽ được lưu trữ tại thư mục hiện hành tức là C:\TP\BIN.

Lệnh Writeln(t1,'Tep van ban'); sẽ ghi vào biến tệp T1 (cũng có nghĩ là ghi vào tệp DULIEU.DAT) dòng chữ "Tep van ban" tiếp đó là các ký tự điều khiển CR, LF.

Lệnh Write(T1,123); sẽ tự động chuyển đổi số 123 thành ký tự và ghi vào tệp không kèm theo CR, LF.

Lệnh Write(T1,' ',123.45); ghi vào tệp một khoảng trống trước dãy 123.45 tiếp đó chuyển đổi và ghi số 123.45 vào tệp theo quy cách ngầm định của Pascal.

Dữ liệu sẽ được ghi vào tệp như sau

Tep van ban

123 1.234500000E+02

(dòng rỗng)

Từ ví dụ trên có thể nêu một số nhận xét sau:

* Các ký tự nhập vào tệp phải nằm trong cặp dấu nháy đơn

* Các số không cần để trong dấu nháy, Pascal sẽ tự động chuyển đổi chúng khi ghi vào tệp. Nếu chúng ta không chỉ rõ quy cách đọc hoặc ghi số thì các số sẽ được ghi theo quy cách chuẩn, cụ thể là:

- Số nguyên sẽ được ghi chính xác như giá trị của nó trong lệnh Write(...)

- Số thực sẽ được chuyển thành dạng khoa học nghĩa là dạng số với luỹ thừa cơ số 10, ví dụ 123.45 chuyển thành 1.234500000E+02. Trong cách viết khoa học Pascal ngầm định như sau:

- Phần nguyên là 1 ký tự

- Phần lẻ là 10 ký tự
- Phần còn lại là 4 ký tự cho luỹ thừa cơ số 10, ví dụ:

$E+02 = 10^2$, $E-03 = 10^{-3}$

* Các số có thể chỉ rõ quy cách ghi như sau

Write(T1,123:5);

Ghi vào tệp số nguyên 123 với độ dài 5 ký tự, Pascal sẽ để trống hai vị trí bên trái Write(T1,' ',123.45:6:2);

Số 123.45 được ghi vào tệp với độ dài 6 ký tự và 2 ký tự chứa số lẻ

4. Đọc dữ liệu từ tệp văn bản

Dữ liệu lưu trữ trong tệp văn bản có thể cho hiện lên màn hình bằng lệnh TYPE của DOS, bằng các phím chức năng F3, F4 của NC hoặc đưa vào Word để xem và sửa chữa nếu cần.

Với Pascal tệp văn bản thuộc loại tệp truy nhập tuần tự, do vậy nếu cố tình dùng lệnh truy nhập ngẫu nhiên Seek(...) thì máy sẽ báo lỗi:

Error 63: Invalid File Type

Sau khi tiến hành mở tệp, con trỏ tệp sẽ được đặt tại dòng đầu. Cách thức mà Pascal dùng để đọc dữ liệu là như sau: dùng thủ tục Read, hoặc Readln để đọc dữ liệu từ dòng hiện thời và gán vào một biến kiểu chuỗi đã được khai bao, viết biến đó ra màn hình hoặc máy in.

Read(Bien_tep, Dong); hoặc Readln(Bien_tep, Dong);

Write(Dong); hoặc Writeln(Dong);

Trong hai ví dụ trên Biến trung gian là Dong, biến này phải được khai báo trước và phải thuộc kiểu String.

Để có thể viết toàn bộ dữ liệu từ một tệp văn bản ra các thiết bị ngoài thì các lệnh đọc viết trên phải được lặp đi lặp lại từ dòng 1 đến dòng cuối cùng nghĩa là chương trình phải sử dụng một trong hai vòng lặp:

```
While not eof(Bien_tep) Do
 Begin
   Readln(Bien_tep, Dong);
   Writeln(Dong);
 End;
Hoặc
 For i := 1 to Filesize(Bien_tep)  Do
    Begin
Readln(Bien_tep, Dong);
    End;
```

Nếu trên một dòng chúng ta đã ghi nhiều giá trị ứng với các biến khác nhau thì không thể đọc riêng từng biến. Cách mà chúng ta ra lệnh cho Pascal ghi dữ liệu vào tệp sẽ ảnh hưởng rất nhiều đến việc chúng ta gọi dữ liệu ra và xử lý dữ liệu đó.

Ví dụ sau đây sẽ giải thích cụ thể điều này.

Ví dụ 9.3

```
Var
   F:text;
   Hoten:string[25]; heso:real;  socon:byte; t:string;
Begin
Hoten:=' Tran Van Tam';   Heso:=4.25;   Socon:= 2;
Assign(f,'hoso.txt');
Rewrite(f);
Witeln(f,hoten,' ',heso:4:2,' ',socon);
 Close(f);
 Reset(f);
 Readln(f,t);
 Writeln(t);
Readln;
End.
```

T	r	a	n		V	a	n		T	a	m		4	.	2	5		2	CR	LF	EOF

Chương trình trên đây có một số điều cần phải lưu ý như sau:

1. Lệnh ghi vào tệp:

Witeln(f,hoten,' ',heso:4:2,' ',socon); sẽ ghi tất cả dữ liệu trên một dòng.

Nhìn vào hình vẽ dễ dàng nhận thấy rằng chuỗi Hoten chiếm 12 bytes, Heso chiếm 5 bytes và Socon chiếm 1 byte. Trong thực tế các số kiểu Real có độ dài là 6 bytes nhưng vì chúng ta đã yêu cầu Pascal ghi Heso dài 4 ký tự trong đó có hai số lẻ nên khi ghi vào tệp, Heso chỉ chiếm 4 bytes. Giữa các biến có hai khoảng trống do đó tổng chiều dài của dòng là 19 bytes.

Lệnh đọc dữ liệu từ tệp Readln(f,t) sẽ đọc cả dòng hiện thời vào biến t và lệnh viết Writeln(t) sẽ đưa ra màn hình cả dòng nghĩa là ta lại nhận được toàn bộ dòng như trên. Cần phân biệt rằng dữ liệu viết ra là các ký tự của bảng mã chứ không phải các chữ và số như ta đã nhập vào.

Trường hợp biến t không đủ độ dài để chứa hết cả dòng thì số ký tự thừa sẽ bị cắt bỏ.

2. Nếu chúng ta viết lệnh đọc:

Readln(f,Hoten,' ',Heso,' ',socon);

thì máy sẽ báo lỗi: Error 106 : Invalid Numeric Format

nghĩa là không thể cùng một lúc đọc từ tệp ra màn hình cả biến kiểu ký tự và biến kiểu số.

3. Nếu chúng ta viết lại lệnh ghi dữ liệu:

Witeln(f,hoten);

Writeln(f,heso:5:2,' ',socon);

nghĩa là ghi riêng dữ liệu kiểu "chữ" trên một dòng còn dữ liệu kiểu "số" trên dòng khác, giữa các số có một khoảngcách, sau đó dùng lệnh đọc:

Readln(f,hoten);

Readln(f,heso, Socon);

thì lại nhận được thông báo lỗi ở dòng Readln(f,heso, Socon);

Error 106 : Invalid Numeric Format

nghĩa là các số ghi vào tệp trên cùng một dòng thì cũng không thể đọc chúng như là các biến.

4. Chúng ta sửa lại lệnh ghi một lần nữa

```
Writeln(f,hoten);
Writeln(f,heso:5:2);
Writeln(f,socon);
```

Rồi dùng lệnh đọc

```
Readln(f,Hoten);
Readln(f,Heso);
Readln(f,socon);
```

sau đó là lệnh viết

```
Writeln(Hoten, ' ', Heso:5:2,' ',socon,' ' ,Heso*290000);
```

thì chương trình sẽ không báo lỗi .

Đến đây có thể rút ra kết luận là:

* Muốn lấy lại kiểu của dữ liệu nhập vào tệp văn bản thì tốt nhất là mỗi biến phải nhập trên một dòng, cũng có thể nhập nhiều biến cùng kiểu trên một dòng.

* Với các biến kiểu số đã ghi riêng rẽ trên một dòng khi gọi ra Pascal sẽ tự động chuyển đổi từ dạng ký tự thành dạng số và ta có thể đưa các số này vào các biểu thức tính toán bình thường. Trong ví dụ trên chúng ta tính Lương bằng cách lấy Heso*290000.

* Trong bộ nhớ của máy dữ liệu được ghi liên tục trong các ô nhớ, để phân biệt các dòng Pascal dùng cặp ký tự điều khiển CR và LF. Nói cách khác dữ liệu được lưu trữ liên tục chứ không phải dưới dạng bảng, khi lấy dữ liệu ra chúng ta cũng lấy liên tục nhưng lại có thể bố trí trên màn hình sao cho trực quan và dễ theo dõi.

Ví dụ 9.4: xây dựng một chương trình đơn giản để quản lý công chức. Dữ liệu nhập vào bao gồm Họ tên, Hệ số lương và Số con. Dữ liệu xuất ra màn hình bao gồm Họ tên, Hệ số lương, Số con và Lương tháng. Lương tháng ở đây tính theo quy định của nhà nước = heso*290000.

Chương trình đặt ra hai khả năng lựa chọn:

a. Nếu tệp dữ liệu đã tồn tại thì nhập thêm người

b. Nếu tệp chưa có thì mở tệp mới

Trong cả hai trường hợp đều yêu cầu cho biết số người cần nhập.

Dữ liệu in ra dưới dạng bảng.

Ví dụ 9.4

```
Program Quan_ly_can_bo;
Uses crt;
Var f:text; hoten:string[20]; c1,heso:real; c2,i,n,socon:byte;  ten:string[12];
  Begin
  clrscr;
  Write('Cho biet ten tep '); readln(ten);
```

```
        {$I-}
        assign(f,ten);
        reset(f);
        {$I+}
         if Ioresult=0 then
         Append(f)
         else
        rewrite(f);
        write('Nhap bao nhieu nguoi '); readln(n);
         for i:= 1 to n do
           Begin
           Write(' Ho ten ' ); Readln(hoten);
           Write( ' He so '); readln(heso);
      Write(' So con ' ); Readln(socon);
      Writeln(f,hoten);
      Writeln(f,heso:4:2);
      writeln(f,socon);
      End;
      close(f);
      assign(f,ten);
      reset(f);
      writeln(':----------------------:------:----------:---------------:');
      writeln('|   Ho va ten       | Hs | socon|   Luong   |');
      writeln(':----------------------:-------:---------:---------------:');
      while not eof(f) do
       begin
      readln(f,hoten);
      readln(f,heso);
       readln(f,socon);
   writeln('|',hoten:19,'|',heso:4:2,'| ',socon:4,' | ',heso*290000:10:2,' |');
        end;
        readln;
        end.
```

Ví dụ 9.4 có sử dụng định hướng chương trình dịch {$I+}, {$I-}. Khi định hướng là {$I+} thì chương trình sẽ kiểm tra lỗi vào ra IO (Input, Output) nếu phát hiện thấy lỗi thì dừng chương trình, đây là chế độ ngầm định của Pascal. Nếu định hướng là {$I-} thì việc kiểm tra lỗi vào ra tạm thời không thực hiện, nghĩa là nếu phát hiện thấy lỗi thì tạm treo các thủ tục vào ra và tìm xem hàm IORESULT cho kết quả là gì. Nếu

hàm này cho kết quả bằng 0 thì có nghĩa là việc kiểm tra IO không có gì sai sót và chương trình tiếp tục làm việc. Nếu hàm Ioresult cho kết quả khác 0 thì có nghĩa là việc kiểm tra IO phát hiện thấy lỗi và chương trình cần phải được sửa chữa.

Ví dụ 9.5

Tạo tệp văn bản đặt tên là Baitho.txt để lưu trữ một bài thơ có n dòng, dòng cuối cùng ghi " Nam 2003"

```
Program tep_van_ban;
Uses crt;
var i,n:integer;  f:text; t:string;
Begin
clrscr;
assign(f,'baitho.txt');    rewrite(f);
writeln('Bai tho gom bao nhieu cau?  '); readln(n);
writeln('Hay nhap bai tho cua ban');
for i:= 1 to n do
begin
write('Cau ',i,'  '); readln(t); writeln(f,t);
end;
writeln(f,'Nam ', 2003);
close(f);
clrscr;
gotoxy(15,5); i:=6;
writeln('Du lieu viet tu tep baitho.txt ');
reset(f);
while not eof(f) do
begin
readln(f,t); gotoxy(15,i);   writeln(t);
i:=i+1;
end;  readln;
End.
```

Ví dụ 9.5 có thể cải tiến theo nhiều cách để có được một chương trình đẹp dùng cho việc lưu trữ văn bản, chẳng hạn chúng ta sẽ đưa vào các chương trình con GHIMOI, GHITHEM và DOC phục vụ việc ghi dữ liệu vào tệp mới, ghi thêm dữ liệu vào tệp đã có hoặc đọc từ tệp ra. Trước khi ghi hoặc đọc cần kiểm tra xem tệp đã tồn tại chưa, khi nhập văn bản vào tệp không hạn chế số dòng, muốn kết thúc việc nhập thì khi bắt đầu một dòng mới chỉ cần bấm dấu "*" v.v... (xem ví dụ 9.6).

Ví dụ 9.6:

```
Program tep_van_ban;
Uses crt;
var i,j,n:integer;  f:text; t:string;
   tl,tl1,tl2:char; ten:string[12];

Procedure Ghimoi(ten:string);
Begin
clrscr;
assign(f,ten);
```

```
rewrite(f);
writeln('Hay nhap bai tho cua ban');
 i:=1;
 repeat
 write('Cau ',i,' '); readln(t);
 if t<>'*' then  writeln(f,t);
 i:=i+1;
 Until t='*';
 close(f);
End;

Procedure Ghithem (ten:string);
Begin
clrscr;
assign(f,ten);
append(f);
writeln('Hay bo sung bai tho cua ban');
 i:=1;
 repeat
 write('Cau ',i,' '); readln(t);
 if t<>'*' then  writeln(f,t);
 i:=i+1;
 Until t='*';
close(f);
end;

  Procedure Doc(tep:string);
  Begin
clrscr;
gotoxy(15,5); i:=6;
writeln('Du lieu viet tu tep ',tep);
assign(f,tep);
reset(f);
while not eof(f) do
begin
readln(f,t); gotoxy(15,i);
writeln(t);
i:=i+1;
end;
readln;
end;

  BEGIN   { Thân chương trình mẹ}
  Clrscr;
  Write('cho biet ten tep '); Readln(ten);
  Write('Ban Ghi hay Doc du lieu G/D '); Readln(tl1);
    If upcase(tl1)='G' then
      Begin
```

```
  write('Ghi tep moi hay ghi them vao tep cu M/C ');
  Readln(tl2);
   If upcase(tl2)='M' then Ghimoi(ten)
   else
   Ghithem(ten);
   End
   Else
   Doc(ten);
END.
```

Trong ví dụ 9.5 có sử dụng một biến toàn cục là Ten, kiểu của biến này là string[12]. Khi lấy Ten làm tham số thực để truyền cho các chương trình con thì cần lưu ý rằng các ***tham số trong chương trình con cũng phải thuộc kiểu String và không được khai báo độ dài***. Cụ thể nếu viết dòng lệnh:

Procedure Doc(tep:string);

Dưới dạng mới là

Procedure Doc(tep:string[12]); thì máy sẽ báo lỗi.

VI. TỆP CÓ KIỂU

Tệp có kiểu là tệp mà mọi phần tử đều có cùng độ dài và cùng kiểu. Kiểu các phần tử của tệp có thể là số nguyên, thực, ký tự, chuỗi, mảng hoặc bản ghi. Cách thức khai báo biến kiểu tệp đã trình bày trong mục II. Sự khác nhau cơ bản giữa tệp có kiểu và tệp văn bản là tệp có kiểu ***có thể vừa ghi vào vừa đọc ra,*** còn với tệp văn bản chúng ta buộc phải kết thúc ghi bằng lệnh Close(bien_tep) thì mới có thể đọc tệp, còn khi đang đọc tệp chúng ta cũng phải kết thúc đọc và đóng tệp thì mới có thể ghi thêm dữ liệu vào tệp.

1. Đọc và ghi

1.1. Ghi lên tệp

Write(bientep, bien1, bien2, ...)

bien1, bien2, ... là các biến cùng kiểu với biến tệp.

1.2. Đọc tệp

Read(Bientep, bien1, bien2, ...)

Đọc tệp thực chất là đọc các phần tử của tệp và gán cho các bien1, bien2,... cùng kiểu. Việc đọc diễn ra trong bộ nhớ do đó người sử dụng muốn biết được các giá trị cụ thể thì phải viết các biến đã đọc lên màn hình.

Chú ý:

Khác với tệp văn bản, việc ghi và đọc tệp có kiểu không sử dụng các lệnh Writeln hoặc Readln nghĩa là tệp có kiểu không ghi dữ liệu thành các dòng. Các phần tử của tệp có kiểu được ghi liên tục trong các ô nhớ và chỉ có ký hiệu kết thúc tệp EOF.

Khi chúng ta đọc hoặc ghi xong một phần tử thì con trỏ tệp sẽ tự động chuyển đến vị trí kế tiếp.

2. Truy nhập vào tệp

Seek(bientep,i); i=0,1,2,.....

Thủ tục Seek sẽ định vị con trỏ tại phần tử thứ i của tệp, Pascal quy định vị trí các phần tử là 0, 1, 2... như vậy phần tử thứ nhất là phần tử có vị trí 0.

Lệnh Seek(bientep, 5); sẽ định vị con trỏ tệp tại phần tử thứ 6.

3. Các hàm xử lý tệp

3.1 Filesize(bientep) cho biết số phần tử có trong tệp

3.2 FilePos(bientep) cho biết vị trí hiện thời của con trỏ tệp

3.3 Eof(Bientep) cho giá trị True nếu con trỏ tệp ở vị trí cuối tệp (vị trí cuối tệp được hiểu là vị trí sau phần tử cuối cùng), nếu con trỏ tệp định vị tại bất kỳ phần tử nào của tệp thì hàm eof() cũng cho giá trị False.

Giả sử tệp DS có 8 phần tử khi đó hàm

Filesize(DS) sẽ cho kết quả là số 8, còn thủ tục

Seek(DS,Filesize(DS)) sẽ định vị con trỏ tại ví trí thứ 8 nghĩa là cuối tệp bởi vì phần tử cuối của tệp ở vào vị trí 7.

Ví dụ 9.7

Tạo một tệp lấy tên là TEPCK.DAT để vừa ghi vừa sửa dữ liệu

```
Program  kieutep;
Uses crt;
Var bt: file of byte; i:byte; n:real;
 Begin
 clrscr;
 assign(bt,'TEPCK.DAT');
 rewrite(bt);
 for i:= 0 to 5 do write(bt,i); {ghi vào tệp năm số nguyên}
 reset(bt);
 writeln('  Du lieu luu tru trong tep TEPCK.DAT ');
 while not eof(BT) do    {viết dữ liệu từ tệp ra màn hình}
 begin
 Read(bt,i); write(i:5);
 end;
 writeln;
 seek(bt,3);  {định vị con trỏ tại phần tử thứ 4}
 textcolor(magenta);
 read(bt,i);   {đọc phần tử thứ 4 vào biến i}
 writeln('So cu trong tep o vi tri 3:   ',i);
 i:=33;
 seek(bt,3);
 write(bt,i);  {sửa phần tử thứ 4, giá trị mới là 33}
seek(bt,3); read(bt,i)
 Writeln('So moi trong tep o vi tri 3: ', i);
 writeln('Vi tri hien thoi cua con tro: ',filepos(bt));
 readln;
close(bt);
End.
```

Nhận xét:

Ví dụ 9.7 cho thấy khi ghi hoặc đọc dữ liệu vào tệp có kiểu thì chúng ta luôn luôn phải dùng đến các biến có cùng kiểu với biến tệp.

Đoạn chương trình

i:=33; seek(bt,3); write(bt,i);

dùng để ghi vào phần tử thứ 4 giá trị mới bằng 33.

Nếu chúng ta viết lại đoạn lệnh này với ý tưởng ghi trực tiếp giá trị 33 vào phần tử thứ 4 seek(bt,3); write(bt,33); thì Pascal sẽ báo lỗi "Variable Indentifier Expertied" nghĩa là Pascal không hiểu số 33 thuộc kiểu dữ liệu gì.

Ví dụ 9.8

Lập chương trình để nhập điểm cho học sinh và ghi kết quả vào tệp có tên là DIEM.DAT. Chương trình định nghĩa một kiểu dữ liệu mới là Bangdiem (Kiểu Record) với các trường: Stt, Hoten, Diachi, Gioitinh, Lop, Toan, Ly, Hoa. Biến tệp Ds thuộc loại tệp có kiểu. Ví dụ 9.7 chỉ mới làm công việc là nhập dữ liệu vào tệp mà chưa đọc dữ liệu ra.

Ví dụ 9.8:

```
Program Kieutep;
Uses Crt;
Type
Bangdiem = Record
   Stt: Integer;
   Hoten, Diachi: String[25];
   Gioitinh, Lop: string[5];
   Toan,Ly,Hoa: Real;
End;
Tepdiem = File of Bangdiem;
Var
DS: Tepdiem;
NGUOI: Bangdiem;
 i,j: Integer;  lam,TL: Char;

Begin
Assign(DS,'diem.dat');
Rewrite(DS);
Nguoi.stt:=1;
textbackground(white);
textcolor(5);
lam:='L'; TL:='C'; While lam='L' do
Begin
clrscr;
i:=10; j:=5;
gotoxy(i,j); write(' So thu tu: ');
gotoxy(i,j+1); write(' Ho va ten: ');
gotoxy(i,j+2); write(' Nam hay nu: ');
```

```
gotoxy(i,j+3); write(' Thuoc lop: ');
gotoxy(i,j+4); write(' Dia chi: ');
gotoxy(i,j+5); write(' Diem Toan: ');
gotoxy(i,j+6); write(' Diem Ly: ');
gotoxy(i,j+7); write(' Diem Hoa: ');
i:=i+15;
With Nguoi do
Begin
Gotoxy(i,j);    Readln(STT);
Gotoxy(i,j+1); Readln(Hoten);
Gotoxy(i,j+2); Readln(Gioitinh);
Gotoxy(i,j+3); Readln(lop);
Gotoxy(i,j+4); Readln(Diachi);
Gotoxy(i,j+5); Readln(Toan);
Gotoxy(i,j+6); Readln(Ly);
Gotoxy(i,j+7); Readln(Hoa);
End;
Gotoxy(i,j+9); Write('Co ghi vao danh sach khong? C/K ');
Readln(TL);
If upcase(TL) ='C' then
Begin
Write(DS,NGUOI);
NGUOI.STT:= NGUOI.STT+1;
End;
textbackground(white);
textcolor(blue);
Gotoxy(28,22);
Write(' NHAP TIEP HAY THOI ? L/T ');
Readln(lam);
lam:=upcase(lam);
textbackground(white);
textcolor(5);
End;
Close(DS);
End.
```

Ví dụ 9.9 Nhập dữ liệu quản lý điểm tuyển sinh trung cấp vào tệp, tính tổng điểm và kết quả cho từng người sau đó đọc dữ liệu ra màn hình.

```
Program Kieutep;
Uses crt;
 Type dong=record
    mhs:byte; hoten:string[12];
    toan,ly,tong:real; ketqua:string[7];
    end;
```

```
ds = file of dong;
Var nguoi:dong; dsl:ds; i,n:byte;
 Begin
 clrscr;
 assign(dsl,'dsl.txt');    rewrite(dsl);
 write('Nhap bao nhieu nguoi ? '); readln(n);
 for i:= 1 to n do
 Begin
 With nguoi do
  Begin
  write('Ma ho so  ', i); mhs:=i;writeln;
  write('Ho va ten '); readln(hoten);
  write('Diem toan '); readln(toan);
  write('Diem ly   '); readln(ly);
  tong:=toan+ly;
  if tong>10 then ketqua:='Do' else ketqua:='Truot';
  end;
 write(dsl,nguoi);     writeln;
 End;
 close(dsl);
 clrscr;
 reset(dsl);
 writeln('      Du lieu luu tru trong tep DSL.TXT ');
 while not eof(dsl) do
 with nguoi do
 begin
 read(dsl,nguoi);
writeln(mhs:3,hoten:15,toan:5:2,ly:5:2,tong:7:2,ketqua:6);
 end;    readln;
 End.
```

BÀI TẬP ỨNG DỤNG CHƯƠNG 9

1. Viết chương trình nhập vào tệp văn bản F một số dòng văn bản, đếm xem trong tệp các ký tự a,b,....z mỗi ký tự xuất hiện bao nhiêu lần?
2. Thông tin về thí sinh thi đại học bao gồm Hoten, namsinh, diemthi, ketqua. Thông tin trên được ghi trong một tệp văn bản, mỗi trường trên một dòng. Lập chương trình in ra màn hình mỗi thí sinh trên một dòng

   Ví dụ:

Trong tệp	In ra màn hình
Tran Tam 1980 8 Kha	Tran Tam 1980 8 Kha

3. Dữ liệu quản lý hàng hoá bao gồm: Mã hàng, Tên hàng, Số lượng, Đơn giá, Ngày nhập, Số hoá đơn. Lập chương trình nhập dữ liệu từ bàn phím và ghi vào tệp có kiểu. Cho biết một số hoá đơn, hiện lên màn hình số liệu ứng với hoá đơn đó.
4. T1 và T2 là các tệp văn bản, nội dung T1, T2 nhập từ bàn phím, hãy nối T1 với T2 thành T3, hiện T3 lên màn hình.
5. T1 là tệp văn bản chứa 15 ký tự dạng số Integer. Lập chương trình chuyển T1 thành tệp số nguyên T2, tính tổng các số và viết kết quả lên màn hình.
6. Giả sử thư mục hiện hành là C:\TP\BIN. Cho biết một tên tệp từ bàn phím, kiểm tra xem trong thư mục hiện hành có tệp đó chưa, nếu có thì hiện nội dung tệp, nếu chưa thì mở tệp mới để ghi dữ liệu.
7. Bảng lương cơ quan bao gồm các cột: Manv (mã nhân viên), Hoten (họ và tên), Heso (hệ số lương), Chucvu (chức vụ), Baohiem (bảo hiểm), Luong (Lương), Linh (lĩnh). Chức vụ bao gồm hai loại: Lanhdao (lãnh đạo) và Nhanvien (nhân viên). Luong = Heso*540000, Baohiem = 5%luong, Linh = Luong+phucap-baohiem, biết rằng nếu là Lanhdao thì phucap = 100000, còn Nhanvien không có phụ cấp.

   Lập chương trình nhập vào n nhân viên, n không cho biết trước. Tính toán các cột theo yêu cầu trên, hiện dữ liệu ra màn hình dưới dạng bảng.

Chương 10

ĐỒ HOẠ

I. KHÁI NIỆM CHUNG

Màn hình máy vi tính hiện nay có nhiều loại khác nhau, thường dùng nhất là loại màn hình VGA (Video Graphic Adapter), đây là loại màn hình có thể dùng ở một trong hai chế độ: chế độ TEXT - hiển thị văn bản và chế độ GRAPHIC - hiển thị đồ hoạ. Các loại màn hình khác như TVGA (top of Vga) hoặc SVGA (Supper Vga) chất lượng cao hơn cho hình ảnh đẹp hơn và giá cũng đắt hơn.

Trong chế độ TEXT màn hình 14 inch được chia thành 25 dòng và 80 cột, nếu viết kín màn hình ta có thể viết được 2000 ký tự. Chúng ta có thể thay đổi chế độ phân giải để viết ra 25 dòng x 40 cột hoặc 132 dòng x 43 cột. Với các màn hình kích thước lớn hơn ví dụ 15 inch thì số dòng sẽ là 50.

Muốn vẽ hình, tô màu các hình ta phải chuyển sang chế độ đồ hoạ, trong chế độ này màn hình được xem là một ma trận điểm, tuỳ thuộc độ phân giải ta có thể có ma trận 640x480 điểm hoặc 1024x720 điểm.... Mỗi điểm trên màn hình được gọi là 1 Pixel tức là một phần tử ảnh (Picture Element), ta có thể hoàn toàn chủ động trong việc thay đổi màu sắc của từng điểm để tạo ra một bức tranh theo ý muốn. Vị trí của mỗi điểm trên màn hình được biểu diễn bởi hai toạ độ: Hoành độ và Tung độ. Gốc toạ độ (0,0) là điểm ở góc trên bên trái màn hình.

Như đã nêu trong chương 1 phần cài đặt Pascal, muốn chuyển sang làm việc ở chế độ đồ hoạ, trong thư mục hiện hành (thư mục chứa chương trình Pascal) phải có các tệp GRAPH.TPU, *.BGI và *.CHR. Lời gọi đơn vị chương trình đồ hoạ phải đặt ở đầu chương trình ngay sau từ khoá PROGRAM như ví dụ 10.1.

Ví dụ 10.1
```
Program Ve_hinh;
Uses GRAPH;
.....
```

Trong phần thân chương trình cần phải đưa vào các thông báo về kiểu màn hình, chế độ đồ hoạ (MODE) tương ứng. Những người làm tin học ứng dụng thường không quan tâm lắm đến các thông số này do vậy dễ lúng túng khi cần khai báo. Để khắc phục nhược điểm đó trong Turbo Pascal đã có sẵn một thủ tục khởi tạo chế độ đồ hoạ là **Initgraph(var GD,GM: Integer, DP:string[n]);** Khi gọi thủ tục này với các tham số hợp lệ Initgraph sẽ tự xác định kiểu màn hình và Mode đồ hoạ tối ưu . Ví dụ 10.2 Là một chương trình vẽ một đường tròn có tâm tại chính giữa màn hình và bán kính là 50 Pixel.

Ví dụ 10.2
```
Program Ve_hinh_tron;
Uses graph;
Var
GD,DM: Integer;
```

```
Begin
GD:= detect;
Initgraph(GD,GM,'C:\TP70\BGI');
If graphresult <> grok then halt(1);
Circle(320,240,50);
Readln;  CloseGraph;
End.
```

Để gọi thủ tục Initgraph cần phải khai báo trước các tham số GD, GM thuộc kiểu Integer (Trong đó GD: Graph Driver - là một số nguyên xác định kiểu màn hình; GM: Graph Mode - cũng là một số nguyên xác định Mode đồ hoạ).

Nếu ngay sau từ khoá Begin của phần thân chương trình chúng ta khai báo GD:= Detect thì Initgraph hiểu là nó phải tự đi xác định kiểu màn hình và Mode đồ hoạ sao cho đạt kết quả tối ưu. Nói chung trừ những trường hợp đặc biệt, chúng ta không nên tự xác định những thông số này làm gì.

Tham số DP (Driver Path) là đường dẫn tới thư mục chứa các tệp điều khiển kiểu màn hình đồ hoạ, trong ví dụ trên được khai là 'C:\TP70\BGI') nghĩa là ổ đĩa C thư mục TP70, còn BGI là đuôi tệp đều khiển. Cần chú ý là toàn bộ các ký tự khai báo đường dẫn phải đặt trong cặp dấu nháy đơn ' ..... '

Dòng thứ 8 trong ví dụ 10.2

If graphresult <> grok then halt(1);

là câu lệnh kiểm tra giá trị của hàm graphresult , hàm này nhận một số giá trị sau:

Giá trị của hàm	Ý nghĩa
Grok	Tốt, không có lỗi
Grnoinitgraph	Không tìm thấy đơn vị đồ hoạ
Grnotdetected	Không có phần cứng đồ hoạ
GrFilenotfound	Không tìm thấy các tệp điều khiển màn hình đồ hoạ
.....	

Như vậy nếu giá trị của hàm khác Grok nghĩa là có một lỗi nào đó thì dừng chương trình bằng lệnh Halt(1), còn nếu tốt thì tiếp tục làm việc.

II. MỘT SỐ THỦ TỤC CƠ BẢN ĐỂ VẼ HÌNH

1. MOVETO(x,y) : Di chuyển con trỏ đến toạ độ x,y

(x là hoành độ, y là tung độ), x,y là các giá trị kiểu Integer, với màn hình VGA 14 inch thì 0<= x <=639, 0<= y <= 479

2. LINETO(x,y) : Vẽ một đường thẳng từ vị trí con trỏ hiện thời tới toạ độ x,y kết thúc quá trình vẽ con trỏ nằm tại toạ độ mới.

Pascal có sẵn hai hàm để xác định toạ độ góc dưới bên phải màn hình đó là Getmaxx và Getmaxy. Để vẽ một đường chéo của màn hình từ toạ độ 0,0 ta có thể viết các lệnh:

MOVETO(0,0);

LINETO(Getmaxx,Getmaxy);

3. LINE(x1,y1,x2,y2) : Thủ tục này vẽ một đường thẳng từ toạ độ x1,y1 đến toạ độ x2,y2.

4. LINEREL(dX,dY) : Vẽ đường thẳng từ vị trí hiện thời (toạ độ x,y) tới toạ độ x+dx, y+dy.

5. CIRCLE(x,y,r) : vẽ đường tròn tâm tại toạ độ x,y bán kính bằng r Pixel

6. PUTPIXEL(x,y, n) : Thủ tục này sẽ vẽ một điểm sáng tại toạ độ x,y với màu là n. Giá trị n lấy trong khoảng 0-15 hoặc viết trực tiếp tên màu theo tiếng Anh.

7. RECTANGLE(x1,y1,x2,y2) : Vẽ hình chữ nhật không có hoa văn nền toạ độ góc trên bên trái là x1,y1 , toạ độ góc dưới bên phải là x2,y2.

8. BAR(x1,y1,x2,y2) : Vẽ một hình chữ nhật có kèm theo hoa văn nền góc trên bên trái có toạ độ x1,y1 góc dưới bên phải có toạ độ x2,y2. Khi dùng kết hợp với thủ tục số 9 sẽ đồng thời cho phép kẻ các vân hoa trên nền và tô màu cho nền.

9. SETFILLSTYLE(n1,n2) : Thủ tục định vân hoa và màu nền cho thủ tục BAR. n1 là một giá trị nguyên với 0<= n1 <= 11 : định kiểu vân hoa

n2 là số hiệu mã màu đã giới thiệu 0<= n2 <= 15

Trong chương trình khi muốn vẽ hình chữ nhật với màu và vân hoa thì cần đưa thủ tục SETFILLSTYLE(n1,n2) vào trước thủ tục BAR. Giá trị của màu và kiểu vân hoa sẽ được giữ cho đến khi ta định nghĩa lại.

III. THIẾT LẬP MÀU ĐỒ HOẠ

Một số thủ tục trình bày trong mục II không có tham số định màu kèm theo. Nếu muốn định màu ta có thể dùng hai thủ tục sau đây :

10. SETCOLOR(n) : Định màu cho các nét vẽ với 0 <= n <= 15

Dưới đây là bảng giá trị n và màu tương ứng

0 Đen	4 Đỏ	8 Xám xẫm	12 Đỏ nhạt
1 Xanh da trời	5 Tím	9 Xanh da trời nhạt	13 Tím nhạt
2 Xanh lá mạ	6 Nâu	10 Xanh lá mạ nhạt	14 Vàng
3 Xanh lơ	7 Xám nhạt	11 Xanh lơ nhạt	15 Trắng

11. SETBKCOLOR(n) : Định màu nền cho nét vẽ, tham số màu n xem trong bảng trên.

IV. VIẾT CHỮ TRONG CHẾ ĐỘ ĐỒ HOẠ

Khi đã chuyển sang làm việc ở chế độ đồ hoạ ta không thể viết chữ bình thường như trong chế độ văn bản. Muốn viết chữ trong các hình vẽ ta sử dụng một số thủ tục sau đây:

12. OUTTEXT(chuỗi) : Thủ tục này sẽ cho hiện chuỗi ký tự tại vị trí con trỏ hiện thời. Chuỗi có thể viết trực tiếp hoặc thông qua biến chuỗi như trong ví dụ 10.3 sau đây:

Ví dụ 10.3
```
Program Viet_chu;
Uses graph;
Var  GD,DM: Integer;   chuviet : string[30]
Begin
GD:= detect;
Initgraph(GD,GM,'C:\TP70\BGI');
If graphresult <> grok then halt(1);
Begin
outtext('cong hoa xa hoi chu nghia ...');
chuviet:='Viet nam dan chu cong hoa';
outtext(chuviet);
closegraph;
end.
```

12. OUTTEXTXY(x,y,chuoi) : thủ tục này sẽ viết ra chuỗi ký tự tại toạ độ x,y.

13. SETTEXTSTYLE(Kiểu chữ, Chiều viết, Kích thước);

Kiểu chữ là một tham số nguyên nhận giá trị trong khoảng 0-4

Chiều viết chỉ nhận 1 trong hai giá trị : 0 nằm ngang; 1 thẳng đứng

Kích thước Là hệ số phóng to chữ có thể chọn từ 0-10

Để chấm dứt chế độ đồ hoạ trở về chế độ văn bản ta dùng thủ tục CLOSEGRAPH. Sau đó muốn quay lại chế độ đồ hoạ ta lại phải gọi lại INITGRAPH.

Chú ý:

Trong một số trường hợp để chuyển nhanh giữa chế độ đồ hoạ và văn bản chúng ta có thể dùng hai thủ tục sau đây:

RESTORECRTMODE; Tạm ngừng chế độ đồ hoạ chuyển sang chế độ văn bản.

SETGRAPHMODE(n); Ngắt chế độ văn bản đã tạo ra bởi Restorecrtmode thiết lập trở lại chế độ đồ hoạ. Tham số n có thể lựa chọn trong khoảng 0-2. Ví dụ 10.4 dưới đây trình bày cách sử dụng các thủ tục này.

Ví dụ 10.4
```
Program  dohoa_text;
uses  crt,graph;
var  gd,gm : integer;
Begin
gd:=detect;
initgraph(gd,gm,'d:\tp5\bgi');
if graphresult<>grok then halt(1);
moveto(0,0); setcolor(5);
lineto(300,300); delay(2500);
circle(400,300,100); delay(1500);
restorecrtmode;  (* Chuyển về chế độ văn bản *)
```

```
gotoxy(20,20);textcolor(9);
write('AAAAAAAA');
readln;
setgraphmode(2);(* Trở về chế độ đồ hoạ với n=2 cho màn hình VEGA*)
setcolor(blue);
circle(100,100,50);
delay(2000);
restorecrtmode; (* Chuyển sang chế độ văn bản lần thứ hai*)
textcolor(3);
gotoxy(20,0);write('NNNNNNNNNNNNNNN');
readln;
closegraph; (* Kết thúc chế độ đồ hoạ*)
End.
```

Việc sử dụng các thủ tục đồ hoạ không có gì phức tạp, với một chút cố gắng bạn có thể vẽ được những hình rất đẹp theo mong muốn. Ví dụ 10.5 dưới đây là một chương trình vẽ đồ thị hình sin. Chạy chương trình ta sẽ thấy ba đường hình sin với các biên độ và màu sắc khác nhau.

```
Ví dụ 10.5
Program   Do_thi_hinh_sin;
uses graph,crt;
const  m=0.1;
Var  t3,t4,t1,n,t2,gd,gm:integer;    t,x,y,z:real;
 Begin
 gd:=detect;
 Initgraph(gd,gm,'d:\tp5\bgi');
 if graphresult<>grok then Halt(1);
 x:=0;    t3:=100;    n:=0;    t2:=10;
  while t2<=600 do
  Begin
 setcolor(green);
 y:=sin(x);
 t1:=round(y*50);
 t3:=round(y*70);
 t4:=round(y*100);
 t1:=200-t1;
 t3:=200-t3;
  t4:=200+t4;
 moveto(10,200);
 lineto(620,200);
 line(10,80,10,300);
 settextstyle(3,0,3);
 outtextxy(610,205,'x');
```

```
settextstyle(3,0,3);
outtextxy(15,75,'y');
settextstyle(4,0,3);
setcolor(5);
outtextxy(200,300,'do thi ham sin(x)');
setcolor(12);
moveto(10,200);
putpixel(t2,t1,11);
putpixel(t2,t3,14);
setcolor(red);
putpixel(t2,t4,random(14));
setcolor(12);
delay(5);
 x:=x+0.07;
 t2:=t2+1;
 end;
 repeat until keypressed;
 t1:=1;
 t2:=200;
 while t1<=220 do
 begin
 line(340,240,round(sqrt(440*440-t1*t1)),t1);
 t1:=t1+1;
 delay(15);
 end;
 repeat until keypressed;
 closegraph;
 End.
```

Chương trình dưới đây (ví dụ 10.6) thiết kế một đồng hồ ba kim , tốc độ chạy của kim giây tuỳ thuộc vào lệnh DELAY(n) , nếu chọn DELAY(1000) thì cứ 1 giây kim giây chuyển một vị trí. Khi nhập chương trình vào máy cần lưu ý khai báo lại đường dẫn đến thư mục chứa các tệp *.BGI

Ví dụ 10.6

```
Program VEDONGHO;
uses crt,graph;
 var
x,y, maxx,maxy, gd,gm,color,i,j,t:integer;
 N:real;
 LAM,TT:CHAR;
 begin
```

```
        gd:=detect;
        initgraph(gd,gm,'c:\tp70\BGI');
        setcolor(5);
        rectangle(30,20,610,450);
        rectangle(31,21,609,449);
        rectangle(32,22,608,448);
        setfillstyle(9,2);
        bar(33,23,607,447);
        setcolor(red);
        setbkcolor(red);
        for i:=1 to 10 do circle(320,240,i);
        setcolor(11);
        setbkcolor(white);
        for i:=11 to 80 do circle(320,240,i);
         setcolor(14);
         setbkcolor(white);
        for i:=80 to 160 do circle(320,240,i);
         setcolor(white);
        for i:=160 to 200 do circle(320,240,i);
        setcolor(11);
        circle(320,240,79);
        circle(320,240,80);
        setcolor(4);
        circle(320,240,159);
        circle(320,240,160);
        settextstyle(3,0,4);
        outtextxy(310,40,'XII');
        outtextxy(405,60,'I');
        outtextxy(470,120,'II');
        outtextxy(490,200,'III');
        outtextxy(480,290,'IV');
        outtextxy(410,370,'V');
        outtextxy(310,400,'VI');
        outtextxy(210,370,'VII');
        outtextxy(135,290,'VIII');
        outtextxy(130,210,'IX');
outtextxy(155,130,'X');
outtextxy(220,60,'XI');
setcolor(blue);
Settextstyle(4,0,5);
outtextxy(230,100,'DIAMON');
setcolor(random(14));
```

```
for i:=1 to 20 do
circle(320,360,i );
settextstyle(1,0,2);
setcolor(5);
outtextxy(200,450,'Copyright by Dr. Duong Xuan Thanh');
   for i:=1 to 20 do
 begin
 setcolor(random(14));
 circle(320,360,i );
 end;
for i:=1 to 20 do
begin
setcolor(random(14));
circle(320,360,i );

end;
for t:=0 to 12 do  {-------- Kim gio --------}
begin
setcolor(12);
moveto(320,240);
setlinestyle(0,0,3);
SetWriteMode(xorput);
linerel(round(110*cos((t*30-89)*pi/180)),round(110*sin((t*30-89)*pi/180)));
 moveto(320,240);
linerel(round(110*cos((t*30-90)*pi/180)),round(110*sin((t*30-90)*pi/180)));
 moveto(320,240);
linerel(round(110*cos((t*30-91)*pi/180)),round(110*sin((t*30-91)*pi/180)));
 moveto(320,240);
linerel(round(110*cos((t*30-92)*pi/180)),round(110*sin((t*30-92)*pi/180)));
  for i:=0 to 60 do  { ---------Kim phut -------}
  begin
  setcolor(12);
  moveto(320,240);
  setlinestyle(0,0,3);
  SetWriteMode(xorput);
  linerel(round(130*cos((i*6-89)*pi/180)),round(130*sin((i*6-89)*pi/180)));
  moveto(320,240);
  linerel(round(130*cos((i*6-90)*pi/180)),round(130*sin((i*6-90)*pi/180)));
  moveto(320,240);
  linerel(round(130*cos((i*6-91)*pi/180)),round(130*sin((i*6-91)*pi/180)));
(*--------Kim giay--------*)
   for j:=0 to 360 do
```

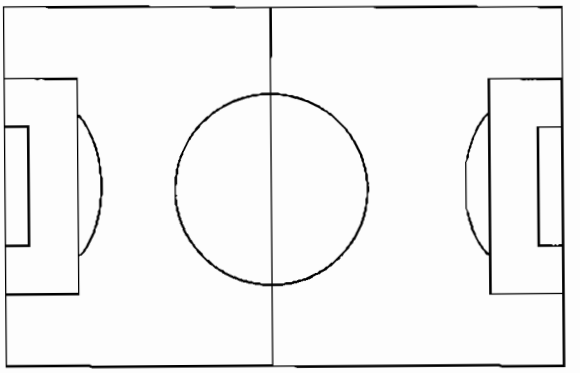
```
begin
moveto(320,240);
setlinestyle(0,0,3);
SetWriteMode(XORPut);
setcolor(12);
linerel(round(150*cos((j-90)*pi/180)),round(150*sin((j-90)*pi/180)));
moveto(320,240);
linerel(round(150*cos((j-91)*pi/180)),round(150*sin((j-91)*pi/180)));
delay(1000);
moveto(320,240);
linerel(round(150*cos((j-90)*pi/180)),round(150*sin((j-90)*pi/180)));
moveto(320,240);
linerel(round(150*cos((j-91)*pi/180)),round(150*sin((j-91)*pi/180)));
end;
moveto(320,240);
linerel(round(130*cos((i*6-89)*pi/180)),round(130*sin((i*6-89)*pi/180)));
moveto(320,240);
linerel(round(130*cos((i*6-90)*pi/180)),round(130*sin((i*6-90)*pi/180)));
moveto(320,240);
linerel(round(130*cos((i*6-91)*pi/180)),round(130*sin((i*6-91)*pi/180)));
end;
   moveto(320,240);
  linerel(round(110*cos((t*30-89)*pi/180)),round(110*sin((t*30-89)*pi/180)));
  moveto(320,240);
  linerel(round(110*cos((t*30-90)*pi/180)),round(110*sin((t*30-90)*pi/180)));
  moveto(320,240);
  linerel(round(110*cos((t*30-91)*pi/180)),round(110*sin((t*30-91)*pi/180)));
  moveto(320,240);
  linerel(round(110*cos((t*30-92)*pi/180)),round(110*sin((t*30-92)*pi/180)));
   end;
  repeat  until keypressed;
  END.
```

BÀI TẬP ỨNG DỤNG CHƯƠNG 10

1. Vẽ lên màn hình lá cờ Tổ quốc.

2. Vẽ hình một chiếc quạt giấy, góc mở lớn nhất của quạt là 90°.

3. Vẽ hình sân vận động theo mẫu sau:



4. Viết dòng chữ " Happy Birth Day " và cho chạy trên màn hình. Kiểu chữ và màu sắc chọn theo quy định của đồ họa.

Chương 11

ÂM THANH

Trong các máy PC thông dụng công suất của loa rất nhỏ do đó việc tạo âm thanh chỉ có tính chất biểu diễn. Muốn tạo ra âm thanh cao thấp khác nhau ta chỉ cần đưa vào loa các xung điện với tần số khác nhau. Turbo Pascal đã có ba thủ tục thiết kế sẵn để làm việc này :

SOUND(n) : tạo ra âm thanh với tần số n , ở đây n phải là một số nguyên dương:

DELAY(n) : Kéo dài tín hiệu âm thanh trong khoảng thời gian n miligiây

NOSOUND : ngắt tín hiệu âm thanh

Cần chú ý rằng khi có thủ tục sound và Delay tín hiệu âm thanh sẽ được phát ra chừng nào chưa có Nosound mặc dù ta đã định khoảng thời gian trễ qua thủ tục Delay.

Để tạo một bản nhạc chúng ta cũng cần có một chút kiến thức về nhạc lý. Độ cao thấp trong âm nhạc được phân thành các quãng tám. Quãng tám trung tần thì nốt Đô có tần số 512 Hz, quãng tám trầm hơn tần số của nốt Đồ sẽ là 256 Hz, còn quãng tám cao hơn nốt Đố có tần số 1024 Hz.

Dưới đây là ví dụ thiết kế bàn phím thành các phím của một chiếc đàn dương cầm. Các nốt đô, rê, mi, fa, son, la , xi đố sẽ bấm các chữ cái tương ứng (D, R, M, F, S, L, X, Z). Muốn dừng âm ta bấm phím P (PAUSE), còn muốn dừng chương trình bấm E (EXIT). Nếu bạn thuộc nhạc của một bản nhạc nào đó xin mời chạy chương trình và gõ các phím theo thiết kế bạn sẽ được nghe bản nhạc do chính mình biểu diễn.

Ví dụ 11.1

```
Program nhac;
uses crt,graph;
var
n :char; i,j:integer;
begin
   clrscr;
   textcolor(14); textbackground(white);
   for i:=1 to 6 do
   begin
   gotoxy(16,i+3);
   for j:=1 to 51 do write(chr(177));
   end;
   gotoxy(17,5);
   textcolor(red);
   write('Do-D, Re-R, Mi-M, Fa-F, Son-S, La-L, Xi-X, Do2-Z');
```

```
   gotoxy(33,6); write(' Re2-W,  Mi2-T ');
   gotoxy(22,7);
   textcolor(blue);
   writeln('   P->Ngat am,  E-> Dung chuong trinh   ');
   gotoxy(22,8);
   textcolor(5);
writeln('Moi ban choi moi ban nhac minh ua thich ');
repeat
n:= readkey;
if n = 'd' then begin nosound; delay(3); sound(523); end;
if n = 'r' then begin nosound; delay(10); sound(587); end;
if n = 'm' then begin  nosound; delay(10); sound(659); end;
if n = 'f' then begin  nosound; delay(10); sound(698); end;
if n = 's' then begin  nosound; delay(10); sound(784); end;
if n = 'l' then begin nosound; delay(10); sound(880); end;
if n = 'x' then begin nosound; delay(10); sound(988); end;
if n = 'z' then begin  nosound; delay(3); sound(1050); end;
if n = 'w' then begin  nosound; delay(3); sound(1190); end;
if n = 't' then begin nosound; delay(3); sound(1300); end;
if n = 'p' then nosound;
until upcase(n) ='E';
nosound;
End.
```

Nếu chúng ta chỉ muốn nghe một bản nhạc phát ra từ máy thì có thể tạo nên một chương trình thiết kế các nốt nhạc sau đó ghép chúng lại thành bản nhạc tuỳ ý.

Trong ví dụ dưới đây các chương trình con T0,T1,T5,T6,T7 nhằm tạo ra thời gian trễ (tức là trường độ của các nốt nhạc). Các chương trình con còn lại tạo nên các nốt nhạc theo tên của chương trình con. Chương trình con LANGTOI ghép các nốt nhạc tạo nên bản nhạc LANGTOI.

Ví dụ 11.2

```
Program Baihat;
uses crt;
   procedure T0;   Begin  delay(2400); Nosound; End;
   procedure T1;   Begin  delay(1600); Nosound; End;
   procedure T5;   Begin  delay(100); Nosound; End;
   procedure T6;   Begin  delay(3200); Nosound; End;
   procedure T7;   Begin  delay(4800); Nosound; End;

   Procedure nt(i:integer);
    Begin  Sound(i); End;
      Procedure si;  Begin nt(680); End;
      Procedure si1; Begin nt(720); End;
```

```
  Procedure do2; Begin nt(390); End;
  Procedure do1; Begin nt(780); End;
  Procedure re1; Begin nt(876); End;
  Procedure re;  Begin nt(441); End;
  Procedure mi1; Begin nt(984); End;
  Procedure mi;  Begin nt(492); End;
  Procedure fa;  Begin nt(519); End;
  Procedure son; Begin nt(586); End;
  Procedure son1; Begin nt(770); End;
  Procedure la;  Begin nt(652); End;

Procedure Langtoi;
 Begin
 clrscr; gotoxy(25,12); textcolor(14); textbackground(red);
 Write(' LANG TOI * Nhac tien chien ');
 Repeat
 do2;t0;mi;t0;son;t6;la;t1;son;t6;t5;
 son;t0;do1;t1;si;t0;la;t0;son;t6;t5;
 la;t0;son;t0;fa;t1;mi;t0;son;t6;t5;
 do2;t0;re;t1;mi;t0;son;t1;do1;t0;re1;t1;mi1;t0;t5;
 re1;t0;do1;t1;re1;t1;do1;t1;son;t0;mi;t0;son;t0;do1;t6;t5;
 do1;t0;do1;t1;la;t6;t5;la;t0;si1;t0;la;t0;son;t6;
 fa;t0;fa;t1;la;t0;t5;la;t1;mi;t0;re;t1;son;t6;
 do2;t0;re;t1;mi;t0;fa;t1;son;t1;mi;t0;re;t6;
 do2;t1;do1;t6;si;t0;re1;t0;son;t6;do1;t7;t1;
 Until keypressed; Nosound;
 End;
BEGIN
 Langtoi;
END.
```

PHẦN 3

HỆ SOẠN THẢO VĂN BẢN MICROSOFT WORD

I. GIỚI THIỆU CHƯƠNG TRÌNH MS-WORD

MS-Word là trình soạn thảo văn bản do hãng MicroSoft thiết kế. MS-Word hoạt động trong môi trường Window. Các máy tính ở nước ta hiện nay vẫn đang dùng hệ điều hành Window 98, Window 2000 và phổ biến nhất là Windows XP.

Chương trình MS-WORD là hệ soạn thảo văn bản nằm trong bộ chương trình Tin học văn phòng (MicroSoft Office), chúng ta có thể gặp các phiên bản khác nhau của Word như Word6.0 trong Office 4.3 hoặc Word 97, Word 2000, Word 2003.... Những phiên bản mới này được bổ sung thêm một số tính năng về đồ hoạ, bảng biểu mà các phiên bản trước không có.

Trong môi trường WINDOWS có thể tìm thấy bảng tính điện tử Excel, hệ quản trị dữ liệu Access, công cụ vẽ Powerpoint và một số trình ứng dụng khác. Trong phần này chúng ta chỉ đề cập đến MS-WORD.

Bộ chương trình Windows được cài đặt trong ổ cứng của máy vi tính. Với phiên bản 3.1 (là phiên bản làm việc trong môi trường DOS) sau khi khởi động từ dấu nhắc hệ thống C:\> ta chỉ cần gõ lệnh WIN và bấm tiếp phím Enter là chương trình được khởi động. Các phiên bản Windows 95 trở đi việc khởi động được hoàn toàn tự động.

Trước khi làm việc với MS-WORD ta cần phải biết cách sử dụng thiết bị chuột (Mouse). Mouse có thể có 2 hoặc 3 phím (Hình 1.1). Khi di chuyển mouse trên bàn làm việc con trỏ mouse trên màn hình sẽ di chuyển theo, tuỳ thuộc vào vị trí của Mouse trên màn hình mà hình dạng của nó sẽ thay đổi. Phần lớn các thao tác của mouse chỉ sử dụng phím trái (Left) do vậy trong tài liệu này nếu nói bấm mouse thì có nghĩa là bấm phím trái, những trường hợp bấm phím phải sẽ có ghi chú riêng. Có ba thao tác bấm phím của chuột :

a. Bấm đơn: đưa chuột đến biểu tượng hoặc vị trí cần thiết rồi bấm phím trái

b. Bấm kép: đưa chuột đến biểu tượng hoặc vị trí cần thiết rồi bấm phím trái hai lần liên tiếp

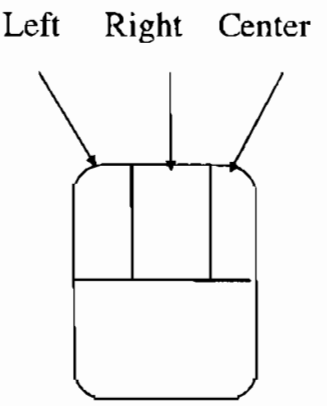
c. Bấm rê : đưa chuột đến vị trí cần thiết bấm và giữ phím trái sau đó di mouse trên bàn, thao tác bấm rê nhằm bôi đen một vùng văn bản hoặc di chuyển một đối tượng từ chỗ này sang chỗ khác.

Hình dạng con trỏ chuột phụ thuộc vào vị trí của nó trên màn hình, mỗi hình dạng có một công dụng khác nhau, chúng ta sẽ thấy trong từng ứng dụng cụ thể.

Một số chuột hiện nay phím giữa được thay bằng một bánh xe. Quay bánh xe này màn hình sẽ được cuộn lên hay xuống.

Chương trình soạn thảo văn bản MS-WORD thường được cài đặt trong thư mục Program Files hoặc thư mục MSOFFICE . Bấm kép phím trái mouse vào biểu tượng MS-WORD là có thể khởi động chúng để làm việc.

Màn hình của chương trình WORD 97 hoặc WORD 2000 được chia làm 2 phần. Phần trên thông thường gồm 4 dải chữ hoặc biểu tượng và được gọi là các thanh (hình 1.2).



Hình 1.1

Thanh tiêu đề : Thanh này cho biết tên cửa sổ đang làm việc (Micrsoft Word) và tên văn bản đang soạn thảo.

Thanh thực đơn (Menu) : Thanh này trình bày các thực đơn ngang, mỗi mục chọn trong thực đơn ngang sẽ cho tiếp một thực đơn dọc.

Thanh công cụ (Toolbars) : Trên thanh công cụ là các nút (Button), các nút này là công cụ giao tiếp thay cho việc chọn các thực đơn con trong thanh Menu.

Thanh định dạng (Format) : Thanh này gồm các nút phục vụ cho việc định dạng văn bản, các chức năng định dạng văn bản thực ra cũng đã có trong thực đơn Format, tuy nhiên định dạng bằng thanh Format sẽ nhanh hơn và tiện dụng hơn.

Có thể dùng chức năng Tools - Customize để thay đổi các nút trong các thanh công cụ hoặc dùng View-Toolbars... để thêm bớt các thanh .

1- Thanh tiêu đề (Title Bar)

Phần giữa thanh tiêu đề là tên của cửa sổ đang mở (Microsoft Word) và tên tệp văn bản đang soạn thảo, khi bắt đầu làm việc với MS-WORD 6.0 tên tệp văn bản được đặt ngầm định là Document1, nếu chúng ta ghi văn bản vào đĩa với tên khác thì tên đó sẽ thay thế cho Document1. Bên trái thanh tiêu đề là nút điều khiển (Ctrol menu box) . Khi bấm mouse vào nút này ta thấy hiện lên một menu dọc gồm các Menu con :

Restore (Khôi phục cửa sổ về trạng thái trước),

Move (dịch chuyển cửa sổ đến vị trí mới),

Size (thay đổi kích thước cửa sổ),

Minimize (Thu cửa sổ thành biểu tượng),

Maximize (Phóng to cửa sổ) ,

Close (Đóng cửa sổ soạn thảo),

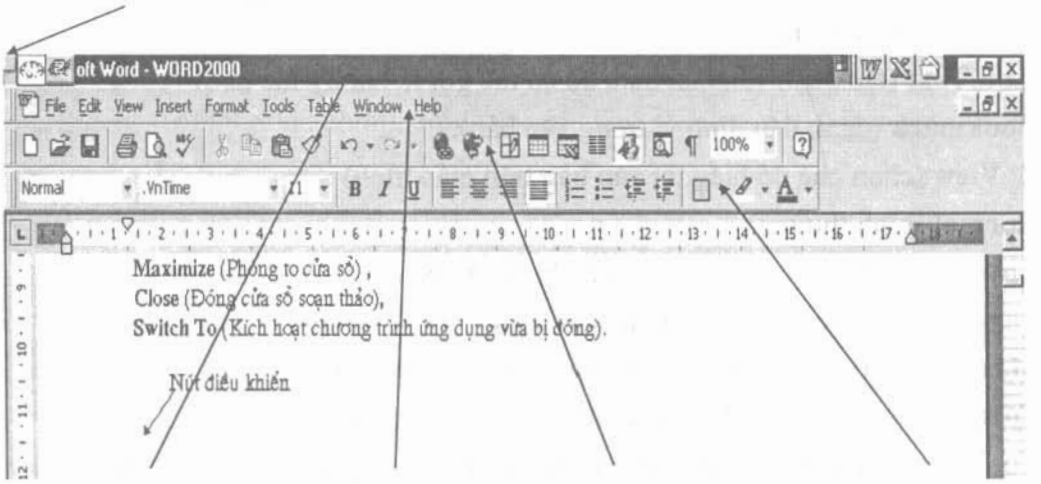
Switch To (Kích hoạt chương trình ứng dụng vừa bị đóng).

Phía bên phải thanh tiêu đề có ba nút

- Thu nhỏ màn hình làm việc thành một dòng
- Thu nhỏ hoặc phóng to cửa sổ
- Kết thúc làm việc, đóng cửa sổ lại

2- Thanh thực đơn (Menu Bar)

Thanh thực đơn gồm một danh mục các thực đơn chính xếp theo hàng ngang. Để chọn các ứng dụng trong một Menu ngang ta bấm đơn phím trái mouse vào menu đó hoặc bấm tổ hợp phím nóng phím (đè phím ALT rồi gõ tiếp chữ cái gạch chân của menu . Mỗi menu ngang lại gồm nhiều mục dọc (Menu popup), để chọn một mục dọc ta làm hoàn toàn tương tự như chọn menu ngang. Các menu ngang và mục dọc tương ứng được liệt kê dưới đây:



Thanh tiêu đề Thanh thực đơn Thanh công cụ Thanh định dạng

Hình 1.2

2.1 File (các lệnh về xử lí tệp) :

New (mở cửa sổ mới để soạn thảo văn bản)

Open (mở văn bản đang lưu trong đĩa ra màn hình)

Save (cất văn bản đang soạn thảo vào tệp trên đĩa)

Save as (cất văn bản vào tệp với tên khác)

Close (đóng văn bản đang soạn thảo)

Find file (tìm tệp trên đĩa)

Summary Information (tạo thông tin tóm tắt về văn bản)

Templates (chọn khuôn mẫu trình bầy văn bản)

Page Setup (định dạng trang văn bản , chọn cỡ giấy, lề in ...)

Print Preview (xem toàn cảnh văn bản trước khi in)

Print (in văn bản).

2.2 Edit (các lệnh phục vụ việc soạn thảo) :

Undo (bỏ thao tác vừa làm)

Repeat (lặp lại thao tác vừa làm)

Cut (xoá đối tượng đã chọn hoặc đoạn văn bản đã bôi đen)

Copy (chép đối tượng đã chọn hoặc đoạn văn bản đã bôi đen vào vùng đệm clipboard)

Paste (dán đối tượng đã có trong vùng đệm vào vị trí mới)

Paste Special (dán đối tượng theo một phương pháp đặc biệt, chuyển cột thành hàng hoặc hàng thành cột, nâng cao mật độ khi in)

Clear (xoá đoạn văn bản đã chọn)

Find (tìm kiếm từ ngữ hoặc đoạn văn trong văn bản)

Replace (tìm và thay thế)

Goto (nhẩy tới trang số..)

AutoText (tạo đoạn văn bản mẫu để có thể gọi ra bất kỳ lúc nào)

Bookmark (đánh dấu định vị trong văn bản).

2.3 View (chọn chế độ hiển thị văn bản trên màn hình) :

Normal (bình thường)

Layout (hiện lề ngoài văn bản hoặc hiện văn bản kèm theo hình vẽ)

PageLayout (hiện lề ngoài trang văn bản)

Master Document (tạo văn bản chính với các văn bản con)

Fulscreen (mở cửa sổ văn bản rộng kín toàn màn hình)

Toolbars.. (cho hiện hoặc không hiện các thanh công cụ, thanh định dạng, thanh đường viền...)

Ruler (cho hiện hoặc không hiện thước kẻ trên đầu cửa sổ văn bản)

Header and Footer (cho hiện tiêu đề đầu và cuối trang)

Footnote (cho hiện chú giải cuối trang)

Annotation (cho hiện chú thích trong văn bản)

Zoom (thay đổi kích thước cửa sổ soạn thảo)

2.4 Insert (chèn các đối tượng khác nhau vào văn bản) :

Break (chèn dấu ngắt trang)

Page Number (đánh số trang)

Annotation (thêm lời chú thích)

Date and Time (ngày và giờ)

Field (chèn mã trường có chứa thông tin xác định vào văn bản) ,

Symbol (chèn các ký tự đặc biệt như α,β... hoặc các dấu hoa văn)

Form Field (biểu mẫu định sẵn)

Foonote (chú giải cuối trang)

Caption (chèn thêm chú giải cho nội dung đã chọn)

Cross reference (chèn các đối tượng qua tham khảo chéo trong hộp liệt kê)

Index and Table (chèn bảng mẫu)

File (chèn tệp)

Frame (chèn khung để đóng gói văn bản hoặc một bức tranh)

Picture (chèn thêm bức tranh đã có sẵn trong thư viện vào văn bản hoặc vào khung)

Objects (chèn các đối tượng khác như bảng tính, các dấu toán học ..)
Database (chèn cơ sở dữ liệu)

2.5 Format (chức năng định dạng văn bản)

Font : (định dạng chữ bao gồm kiểu chữ: (Font) ; kích thước: (Font size); màu sắc: (Color) ; dáng chữ: (Style))

Pragraph (định dạng đoạn văn bản : khoảng cách dòng, độ thụt dòng, lề, khoảng cách giữa các đoạn văn bản)

Tabs (định dạng bước nhảy cột khi gõ phím Tab)

Borders and Shading (định dạng đường bao và bóng của khung, bảng)

Columns (phân chia đoạn văn bản hoặc trang giấy thành các cột ...)

Change-Case (biến chữ to thành nhỏ và ngược lại, định dạng nhóm từ kiểu tên riêng ...)

Drop cap ...(định dạng chữ cái đầu tiên của một đoạn văn bản)

Bullets and Numbering (định dạng nét gạch và đánh số đầu mục)

Heading Numbering (đánh số các tiêu đề)

Autoformat (thiết lập chế độ định dạng tự động)

Style gallery (định dạng theo mẫu đã được thiết kế của Windows)

Styles (lựa chọn hoặc tổ chức kiểu trình bầy có sẵn cho đoạn văn hoặc cho ký tự)

Frame (định dạng khung bao)

Picture (định dạng lại các hình ảnh đã chèn vào văn bản)

Drawing object (định dạng các đối tượng vẽ)

2.6 Tools (Các công cụ trợ giúp)

Spelling (kiểm tra lỗi chính tả tiếng Anh)

Grammar (chỉnh lý văn phạm)

Thesaurus (tìm từ đồng nghiã)

Hyphenation (đặt gạch nối các từ một cách tự động hay bằng tay)

Language (chọn ngôn ngữ viết văn bản)

Word count (đếm số chữ , số từ , số dòng , số đoạn có trong đoạn văn đã lựa chọn)

Autocorrect (thiết lập tên cho một cụm từ hoặc tiêu đề mà nội dung của nó được tự động chỉnh lỗi chính tả)

Mail Merge (trộn dữ liệu vào văn bản chính, dữ liệu có thể tạo ra trong Word hoặc các phần mềm khác như Foxpro, Lotus, Foxbase..., nhằm mục đích tạo ra một văn bản mới)

Envelopes and labels (tạo bao thư cho văn bản)

Protect Document (bảo vệ tài liệu)

Revions (hiệu đính)

Macro (tạo lập các vĩ lệnh, có thể hiểu Macro là một tập hợp tất cả các lệnh được gõ từ bàn phím trừ các thao tác chuột)

Customize (tạo các nút công cụ hoặc menu riêng)

Options (các phương án lựa chọn tổng hợp trình bày màn hình hoặc văn bản)

2.7 Table (thực đơn về tạo lập bảng)

Insert table (chèn thêm một bảng vào vị trí con trỏ)

Delete cells (xoá các ô đã bôi đen)

Merge cells (liên thông các ô đứng gần nhau theo hàng ngang thành một ô)

Split cells (phân chia một nhóm ô thành nhiều ô)

Select row (chọn hàng)

Select column (chọn cột)

Select table (chọn bảng)

Table Autoformat (tự động định dạng bảng theo khuôn có sẵn)

Cell Height and Width (định kích thước của ô hoặc hàng, cột)

Headings (nhập tiêu đề cho cột)

Convert Text to Table (chuyển văn bản dưới dạng bảng thành bảng dữ liệu. Điều kiện cần là các dữ liệu phải ngăn cách nhau bằng dấu phẩy .

Sort Text (xếp bảng dữ liệu theo cột số hoặc cột chữ)

Formula (tính toán với các số liệu có trong bảng)

Split table (phân chia bảng thành 2 bảng)

Gridlines (cho hiện hoặc không cho hiện lưới kẻ bảng)

2.8 Window (các lệnh xử lý cửa sổ văn bản)

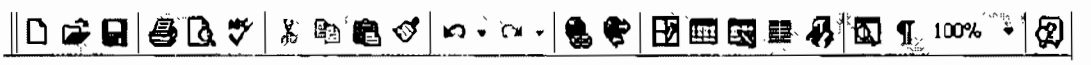
New (mở cửa sổ văn bản mới)

Arrange all (cho hiện đồng thời trên màn hình cửa sổ của các văn bản đang mở)

Split chia cửa sổ hiện hành thành hai phần

2.9 Help (xem hướng dẫn các lệnh)

3- Thanh công cụ (Tools Bar) (hình 1.3)



Hình 1.3

Các nút trên thanh công cụ tính từ trái qua phải gồm:

New (mở một màn hình mới che lên màn hình đang làm việc)

Open (mở một văn bản đang lưu trữ trong đĩa của máy)

Save (ghi văn bản hiện có trên màn hình vào đĩa)

Print (in văn bản ra giấy)

Print Preview (quan sát toàn cảnh văn bản trước khi in)

Spenlling and Grammar (dò lỗi chính tả và ngữ pháp tiếng Anh)

Cut (xoá phần văn bản đang được bôi đen trên màn hình)

Copy (sao chép phần đang bôi đen vào Clipad (vùng đệm bộ nhớ)

Paste (dán những gì đang có trong Clipad vào vị trí hiện thời trên màn hình)

Format Painter (sao chép định dạng của một đoạn văn bản)

Undo Typing (bỏ thao tác vừa thực hiện (khôi phục lại tình trạng trước đó)

Redo cut (khôi phục lại những gì vừa bị xoá)

Isert Hyperlink (thêm siêu liên kết)

Web Toolbar (hiện thanh công cụ kết nối Internet)

Tables and Border (tạo bảng biểu bằng bút vẽ)

Insert Table (tạo bảng biểu)

Isnert Microsoft Excel Worksheet (chèn vào văn bản một trang tính Excel)

Columns (chia văn bản thành nhiều cột)

Drawing (hiện thanh công cụ vẽ)

Document Map (hiện sơ đồ các đề mục của văn bản)

Show/Hide (hiện hoặc tắt ký hiệu xuống dòng)

Zoom (thay đổi tỷ lệ kích thước màn hình)

Office Assistant (trợ giúp)

Thanh công cụ chứa các biểu tượng dưới dạng các nút giúp lựa chọn nhanh một chức năng nào đó. Thanh công cụ chuẩn của Office 97 hoặc Office 2000 (Standard Tools bar) gồm các nút như trong hình 1.3.

Các nút trong thanh công cụ được sử dụng bằng cách đưa chuột đến nút rồi bấm đơn, thao tác này cũng tương tự như chọn một chức năng trong thanh thực đơn rồi chọn chức năng con có tên tương ứng. Ví dụ để ghi văn bản đang có trên màn hình vào đĩa ta chỉ việc bấm chuột vào nút Save thay vì phải chọn thực đơn File rồi thực đơn con Save trong Menu File.

4- Thanh định dạng (Formating Bar) (hình 1.4)

Hình 1.4

Thanh này có các nút dùng để định dạng văn bản, tính từ trái qua phải bao gồm:

Style (kiểu trình bày)
Font (chọn kiểu chữ)
Font size (chọn cỡ chữ)
Bold (chữ đậm)
Italic (chữ nghiêng)
Under Line (gạch chân)
Align Left (dóng thẳng lề trái)
Align Right (dóng thẳng lề phải)
Justify (dãn đều 2 lề)
Numbering (đánh số thứ tự các đoạn văn bản)
Bullets (dùng các biểu tượng đánh dấu đoạn văn bản)
Decrease Indent (dịch chuyển cả đoạn văn bản sang trái)
Increase Indent (dịch chuyển cả đoạn văn bản sang phải)
Border (định dạng đường viền khung)
Highlight (chọn màu nền văn bản)
Font Color (chọn màu chữ văn bản)

II- CÁC PHÍM GÕ TẮT *(ShortCut-Key)*

Các phím gõ tắt thường là một tổ hợp phím ngầm định của Windows hoặc các phím do người sử dụng cài đặt để thực hiện nhanh một công việc nào đó thông qua việc gõ bằng bàn phím . Các phím gõ tắt bao giờ cũng bắt đầu bằng một trong các phím chức năng phụ Ctrl, Shift, Alt kết hợp với một hoặc hai phím khác. Dưới đây là một bảng liệt kê các phím gõ tắt có sẵn của Windows và chức năng tương ứng của nó. Cách bấm phím gõ tắt như sau: Đè phím thứ nhất rồi bấm tiếp phím thứ hai hoặc đè phím thứ nhất, đè tiếp phím thứ hai rồi tiếp phím thứ ba sau đó buông tất cả ra.

Ví dụ khi ghi tổ hợp phím Alt + F thì có nghĩa là đè phím Alt sau đó gõ phím F, còn nếu ghi Ctrl+Shift+W thì có nghĩa là đè hai phím Ctrl và Shift sau đó gõ phím W rồi buông tay ra.

Các phím gõ tắt bắt đầu bằng phím Alt

Alt+F	Chọn thực đơn File trên thanh Menu
Alt+E	Chọn thực đơn Edit trên thanh Menu
Alt+V	Chọn thực đơn View trên thanh Menu
Alt+I	Chọn thực đơn Insert trên thanh Menu
Alt+O	Chọn thực đơn Format trên thanh Menu
Alt+T	Chọn thực đơn Tools trên thanh Menu
Alt+A	Chọn thực đơn Table trên thanh Menu
Alt+W	Chọn thực đơn Window trên thanh Menu
Alt+H	Chọn thực đơn Help trên thanh Menu
Alt+N	Chọn thực đơn Font trên thanh Menu

**** Các phím gõ tắt bắt đầu bằng phím Ctrl***

Ctrl+A	Bôi đen toàn bộ văn bản
Ctrl+C	Sao chép đoạn văn bản đã bôi đen vào Cliboard
Ctrl+V	Dán đoạn văn bản đã có trong Cliboard vào vị trí mới
Ctrl+B	In đậm, chữ béo phần văn bản đã bôi đen
Ctrl+I	In nghiêng phần văn bản đã bôi đen
Ctrl+U	Chữ có gạch chân phần văn bản đã bôi đen
Ctrl+X	Xóa phần bôi đen và lưu vào bộ nhớ
Ctrl+Shift+W	Gạch chân từng từ phần văn bản đã bôi đen
Ctrl+Shift+D	Gạch chân bằng nét kép phần văn bản đã bôi đen
Ctrl+ =	Viết chỉ số dưới (x_1, m_2,....)
Ctrl+Shift+=	Viết chỉ số trên, số mũ (a^2, b^{sinx},...)
Ctrl+ Shift+K	Chữ in hoa nhỏ
Ctrl+ Shift+A	Tất cả chữ in hoa
Ctrl+ Shift+H	Cho ẩn văn bản
Ctrl+ Shift+C	Copy định dạng
Ctrl+ Shift+V	Dán kiểu định dạng
Ctrl+Spacebar	Loại bỏ định dạng

**** Các phím gõ tắt bắt đầu bằng phím Shift***

Shift+End	Bôi đen đoạn văn bản từ vị trí hiện thời đến cuối dòng
Shift+Home	Bôi đen đoạn văn bản từ vị trí hiện thời đến đầu dòng
Shift+ ↓	Bôi đen một dòng văn bản
Shift+ ← (→)	Bôi đen một ký tự bên trái (hoặc bên phải con trỏ)

III. CÁC THAO TÁC ĐỊNH DẠNG

1. Phương pháp viết tiếng Việt

Hiện có hai phương pháp viết tiếng Việt đang được sử dụng là phương pháp viết kiểu TELEX và phương pháp viết kiểu Đánh máy. Trong phạm vi chương trình chúng ta chỉ học phương pháp TELEX. Bộ phông chữ tiếng Việt sử dụng trong chương trình này là bộ phông VietKey, hiện nay ở nước ta còn tồn tại nhiều phông chữ như ABC, Freecode, Vietwear, Vni,... Với các máy kết nối Internet thì phải dùng bộ phông Unicode.

- Cách gõ chữ Việt theo kiểu Telex:

Kiểu Telex là cách gõ tiếng Việt thuận tiện và dễ nhớ, đảm bảo có được tiếng Việt đúng chính tả, đồng thời bỏ đi những cách đặt dấu sai đã tồn tại thành thói quen.

Cách gõ các ký tự đặc biệt của tiếng việt như sau:

Gõ vào	Nhận được	Gõ vào	Nhận được
aw	ă	Aw	Ă
ow	ơ	Ow	Ơ

uw	ư	Uw	Ư
aa	â	AA	Â
oo	ô	OO	Ô
ee	ê	EE	Ê
dd	đ	DD	Đ

Gõ dấu: **f: huyền ; s: sắc ; r: hỏi ; x: ngã ; j: nặng**

Ví dụ : để có dòng chữ " Trường Đại học Chu Văn An" ta phải gõ như sau:

Truwowngf DDaij hocj Chu Vawn An

Lưu ý :

a/ Dấu phải gõ vào cuối chữ , nghĩa là viết xong chữ rồi mới đánh dấu.

Ví dụ: Truwowngf -- Trường, DDieenj -- Điện, baos -- báo

b/ Phím xoá dấu : khi muốn xoá dấu đã đánh trên một từ ta đưa con trỏ đến sát ký tự cuối cùng của từ và gõ phím z.

c/ Chữ ư và ơ hay đi liền nhau nên chúng còn được bố trí ở phím] và [, đây là hai phím liền nhau trên bàn phím kiểu QWERTY, vì vậy gõ sẽ nhanh hơn.

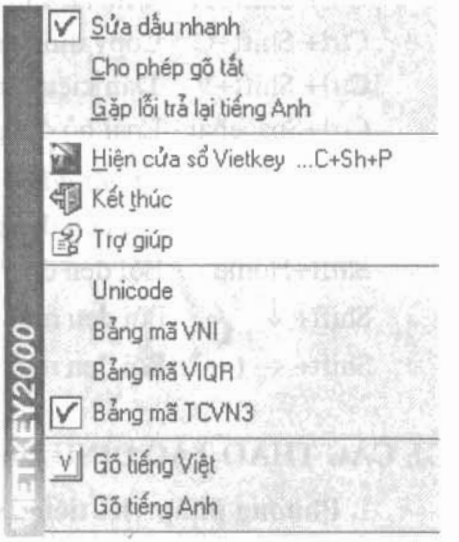
] -- ư [-- ơ } -- Ư { -- Ơ

Thí dụ : tr][ngf -- Trường, dd][cj -- được

- Gõ ooo nhận được oo như cooongs -- coóng

- Nếu bạn muốn gõ dấu [và] thì bạn chỉ việc gõ 2 lần phím đó.

Khi gõ Telex, Vietkey sẽ tự động bỏ dấu đúng chính tả tiếng Việt. Nếu bạn gõ nhầm dấu, bạn cứ việc gõ dấu mới vào mà không phải mất công xoá dấu cũ đi.



Cũng cần lưu ý thêm rằng khi gõ sai dấu trên một từ và chúng ta đã chuyển sang gõ từ khác thì không quay về sửa dấu được mà phải xoá hết nguyên âm của từ sai rồi mới gõ lại từ đó. Để tránh điều phiền toái này bạn có thể chọn tính năng sửa dấu nhanh của Vietkey. Cách thức chọn như sau:

Bấm đơn phím phải chuột vào biểu tượng Vietkey, trên màn hình xuất (hình 1.5) hiện hộp thoại.

Chọn chức năng "Sửa dấu nhanh" bằng cách bấm đơn vào chức năng này. Kể từ nay khi quay lại từ đánh dấu sai ta chỉ việc gõ lại dấu đúng.

2. Định dạng ký tự

2.1. Sử dụng thanh thực đơn

- Bôi đen đoạn văn bản muốn định dạng

- Chọn chức năng **Format**

- Trong Format chọn **Font**, hộp thoại Font xuất hiện (hình 1.6). Trong hộp này có ba mục chọn là **Font** (kiểu chữ) và **Character Spacing** (vị trí chữ) và Text Effects (Hiệu quả trên văn bản).

Hình 1.6

+ **Font** : kiểu chữ, ngay phía dưới là các kiểu chữ có thể lựa chọn. Với bộ phông chữ tiếng Việt chuẩn Vietkey (phông bắt đầu bằng hai ký tự Vn) nếu tận cùng là chữ H thì là phông chữ in.

+ **Font Style** : dáng chữ, có các dáng Regular, Bold, Italic, Bold Italic. Chọn một trong các dáng này và quan sát khung Preview để xem hiệu quả.

+ **Size** : kích thước chữ, có thể chọn kích thước từ 8 đến 72, ngầm định là 12.

+ **Underline** : chữ có gạch chân

+ **Color** : chọn màu cho chữ

+ **Effect** : Một số thao tác định dạng khác:

- **Strikethrough** : tạo nét gạch xuyên qua từ
- **Supescript** : đưa chữ lên cao, tạo số mũ
- **Subscript** : đưa chữ xuống dưới, tạo chỉ số dưới
- **Hidden** : cho ẩn chữ , muốn cho hiện lại bấm tổ hợp phím Ctrl+Z
- **Small Caps** : biến chữ hoa thành chữ thường
- **All Caps** : biến tất cả thành chữ hoa

+ **Preview** : quan sát hiệu quả định dạng

- Trong mục chọn Character Spacing có các nút định dạng như sau:

+ **Spacing** : cách thức hiện chữ trên dòng, có 3 khả năng lựa chọn:

- **Normal** : hiện bình thường
- **Expanded** : các ký tự viết dãn cách nhau
- **Condensed** : các ký tự viết sát vào nhau

+ **Position** : vị trí của ký tự trên đòng

- **Normal** : hiện bình thường
- **Raised** : đưa chữ lên cao hơn
- **Lowered** : đưa chữ xuống thấp hơn

Sau khi lựa chọn xong các đặc tính cần thiết cho chữ bấm chuột vào OK hoặc bấm phím Enter để quay về màn hình soạn thảo văn bản.

2.2. Sử dụng thanh định dạng (Formating)

Trong trường hợp chỉ cần định dạng kiểu chữ, kích thước chữ, và các dáng đậm, nghiêng, gạch chân ta có thể dùng các nút trên thanh định dạng. Phương pháp tiến hành như sau: bôi đen đoạn văn bản cần định dạng, sau đó muốn có chữ đậm chỉ việc bấm chuột vào nút **B** , cần chữ nghiêng bấm nút *I* , cần chữ gạch chân bấm nút U , muốn đưa đoạn văn bản vào giữa trang giấy bấm tiếp nút , còn muốn văn bản dãn đều hai bên mép giấy thì bấm nút ...

Lưu ý : * *Việc chọn màu cho ký tự trong trường hợp a chỉ có ý nghĩa khi quan sát trên màn hình nếu sử dụng máy in mực đen thì cần chú ý vì các màu sáng khi in ra sẽ không rõ.*

** Văn bản hiện có trên màn hình và văn bản khi in ra giấy có thể không giống nhau. Muốn xem văn bản chính thức in trên giấy ta phải chọn chức năng quan sát toàn cảnh Print Preview tức là chọn nút . Những gì quan sát được qua Print Preview thì khi in ra giấy sẽ giữ nguyên 100%.*

2.3. Sao chép định dạng

Sau khi đã làm các thủ tục định dạng một đoạn văn bản ta có thể sao chép sự định dạng này sang các đoạn văn bản khác, thao tác như sau:

- Bôi đen đoạn văn bản đã định dạng

- Chọn nút Format Painter , di chuột vào phần màn hình chứa văn bản khi đó chuột sẽ có thêm một chiếc chổi bên trái.

- Bấm và rê chuột trên đoạn văn bản cần định dạng sau đó buông tay ra, đoạn văn bản sẽ được định dạng theo cách của đoạn ban đầu.

Nếu muốn sao chép định dạng cho nhiều đoạn văn bản cùng một lúc, ta bấm kép vào nút Format Painter rồi lặp lại thao tác như trong bước trên, khi nào sao chép xong bấm đơn lần nữa vào nút Format Painter .

3. Định dạng đoạn văn bản

Đoạn văn bản (Paragraph) được hiểu là toàn bộ những từ nằm giữa hai lần bấm phím Enter.

Thao tác định dạng đoạn văn bản có thể làm trước hoặc sau khi viết văn bản. Nếu ta đã viết xong văn bản rồi mới định dạng thì phải bấm Ctrl+A để bôi đen toàn bộ văn bản.

Các bước tiến hành :

* Bôi đen văn bản hoặc một số đoạn muốn định dạng

* Chọn nút Format - chọn tiếp Paragraph , cửa sổ Paragraph xuất hiện (hình 1.7). Các lựa chọn soorong cửa sổ bao gồm

+ **Indents and Spacing** : vị trí đoạn văn bản

+ **Line and Pagr Break** : sự liên kết các đoạn và ngắt trang

Trong mục Indents and Spacing có các thông số lựa chọn sau:

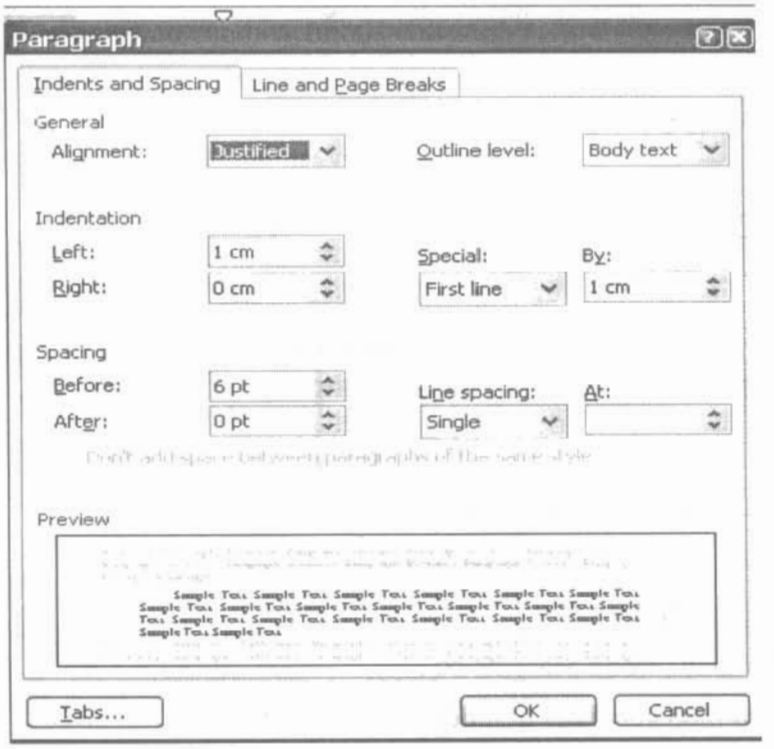
Left : khoảng cách từ đoạn văn bản đến lề trái

Right : khoảng cách từ đoạn văn bản đến lề phải

Before : đoạn hiện thời cách đoạn trên bao nhiêu

After : đoạn hiện thời cách đoạn dưới bao nhiêu

Special : trong Special có hai chức năng



Hình 1.7

- **First line** : vị trí của dòng đầu tiên viết lùi vào bao nhiêu
- **Hanging** : các dòng còn lại viết lùi vào bao nhiêu
- **Line Spacing** : khoảng cách giữa các dòng, bao gồm các lựa chọn
- **Single** : cách nhau bình thường theo kích thước chuẩn
- **1.5 Line** : các dòng cách nhau gấp rưỡi
- **Double** : các dòng cách nhau gấp đôi
- **At least** hoặc **Exactly**: các dòng cách nhau theo lựa chọn của người sử dụng
- **Multiple**: cách gấp 3 so với bình thường (chữ sẽ bị mất phần phía trên)

4. Định dạng trang giấy

Định dạng trang giấy có thể làm trước hoặc sau khi viết văn bản , nói chung nên làm trước. Thao tác định dạng trang giấy bao gồm :

Chọn chức năng File chọn tiếp Page Setup. Hộp thoại Page Setup hiện lên (hình 1.8) với các khai báo :

Hình 1.8

* **Margins** : lề trang giấy , trong Margins có :

Top : khoảng cách từ mép giấy phía trên đến vị trí dòng đầu tiên

Bottom : khoảng cách từ mép giấy phía dưới đến vị trí dòng cuối cùng

Left : lề bên trái

Right : lề bên phải

Gutter : khoảng cách giữa các cột nếu trang giấy chia thành nhiều cột

From Edge : khoảng cách tính từ mép giấy, bao gồm:

Header : khoảng cách từ mép giấy phía trên đến vị trí viết tiêu đề hoặc số trang

Footer: khoảng cách từ mép giấy phía dưới đến vị trí viết dòng chú thích ở đáy trang

* **Paper**: kích thước trang giấy

Trong mục chọn này có thể chọn các cỡ giấy chuẩn : A4, A3, Letter .. hoặc tự định kích thước trong các mục

Width : chiều rộng

Height : chiều cao trang giấy

Orientation : định hướng in :

Portrait : in văn bản theo chiều dọc trang giấy

Landscape : in văn bản theo chiều ngang trang giấy

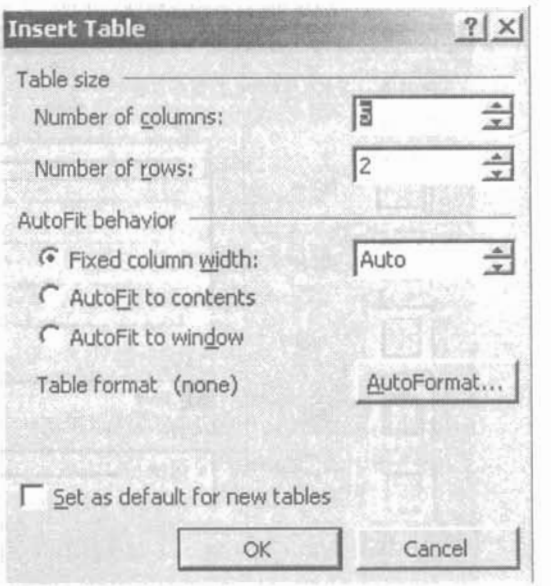
* **Layout:** cách thức chọn trang chẵn, lẻ, có đánh số trên trang đầu tiên hay không…

IV. KẺ BẢNG BIỂU

1. Chèn bảng biểu vào văn bản

Có ba phương pháp đưa vào văn bản một bảng biểu là dùng chức năng **Table** trên thanh thực đơn hoặc dùng nút **Insert Table** trên thanh công cụ hoặc là dùng bút chì và tẩy để vẽ bảng.

Đưa con trỏ đến vị trí cần chèn bảng biểu, bấm đơn vào chức năng Table rồi chọn tiếp Insert - Table hộp thoại Insert Table hiện lên như hình 1.9



Hình 1.9

Mục **Number of Columns** cho phép chọn số cột của bảng biểu, bấm vào các mũi tên phía bên phải con số ngầm định (số 2) để tăng hoặc giảm số cột.

Mục **Number of Rows** cho phép chọn số hàng của bảng biểu, bấm vào các mũi tên phía bên phải con số ngầm định (số 2) để tăng hoặc giảm số hàng.

Mục **Fixed Columns Withs** cho phép chọn độ rộng của các cột trong bảng biểu tính theo Inch hoặc cm, nếu để ngầm định Auto thì MS-WORD sẽ tự động chọn các cột có độ rộng bằng nhau và bằng chiều rộng trang giấy chia cho số cột đã chọn.

Mục **AutoFit to conterts** quy định cột sẽ tự động nới rộng theo nội dung văn bản viết trong cột.

Mục **AutoFit to window sẽ tự động định độ rộng cột theo bề rộng cửa sổ mà bảng biểu sẽ hiện lên.**

Mục AutoFormat dùng để chọn dạng bảng biểu đã thiết kế sẵn trong Word. MS-WORD 2003 đã thiết kế 48 kiểu bảng biểu gọi là 48 style. Các style này chủ yếu khác nhau về các dòng tiêu đề phía đỉnh của bảng biểu. Ví dụ bảng biểu trong hình 1.10 là dạng chuẩn đầu tiên, dạng này có hiệu ứng không gian ba chiều.

	Jan	Feb	Mar	Total
East	7	7	5	19
West	6	4	7	17
South	8	7	9	24
Total	21	18	21	60

Hình 1.10

Bảng biểu đã lựa chọn trong cửa sổ Insert Table sẽ hiện lên tại vị trí con trỏ hiện thời trong văn bản, đó mới là bảng biểu tượng trưng, khi in ra giấy sẽ không có các đường kẻ. Muốn hiện các đường kẻ ta phải bôi đen toàn bộ bảng và chọn chức năng **Format - Borders and Shading.** Hộp thoại Table Borders and shading hiện lên như hình 1.11

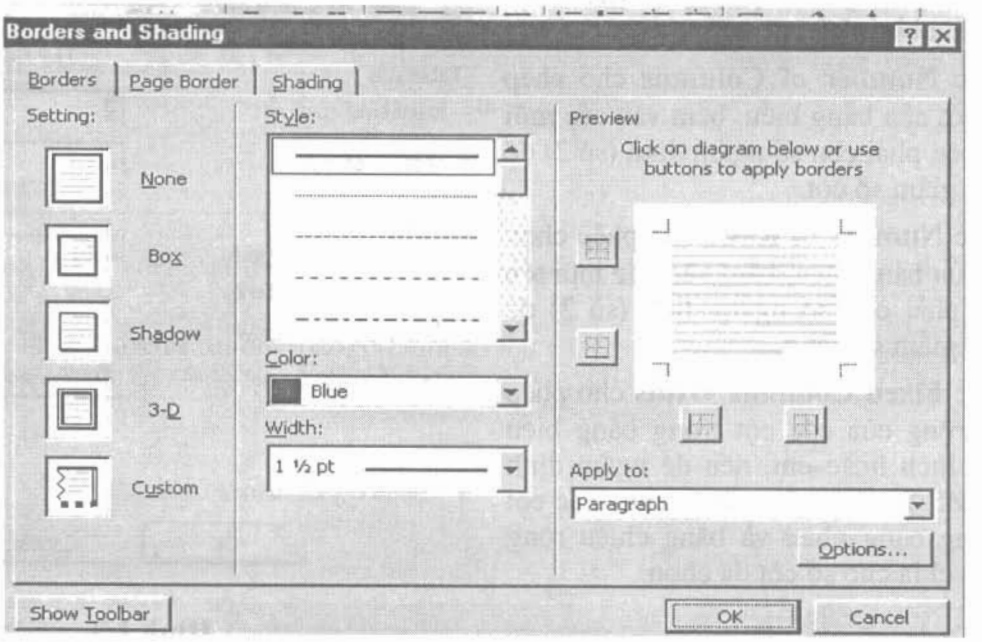
Phần **Bordrs** (đường bao) có một số lựa chọn:

None : không tạo các đường kẻ

Box : chỉ tạo đường bao xung quanh bảng

Shadow : tạo bóng đen phía dưới và bên phải bảng

3-D : hiện bảng dưới dạng hộp không gian ba chiều



Hình 1.11

Custom: tự chọn các nét vẽ bảng.

Để chọn một nét vẽ nào đó cho bảng, ta chọn nét trước ở mục Style, màu sắc nét vẽ chon ở mục Color, độ dày mảnh của nét vẽ chọn ở mục Width, sau khi đã chọn xong thì bấm chuột vào nét vẽ tượng trưng ở mục Preview rồi chọn OK.

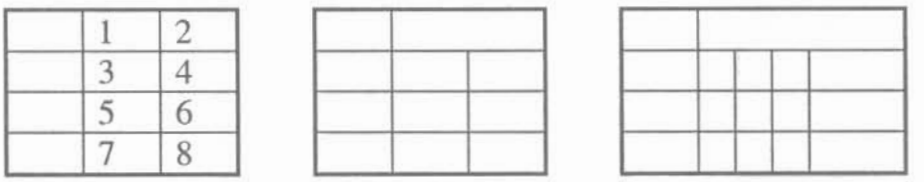
Page Border: Tạo đường viền cho trang văn bản

Shading : chức năng shading dùng để tạo hoa văn nền cho vùng ô đã bôi đen trong bảng biểu, đồng thời với việc chọn hoa văn còn có thể chọn màu cho hoa văn song với các máy in thông dụng hiện nay ta chỉ có thể in đen trắng.Trong cửa sổ Table Borders and shading ta thấy có nut **Show Toolbar**. Nút này dùng để hiện lên thanh công cụ Borders. Sử dụng các nút trên thanh Borders ta cũng có thể tạo nên các đường viền theo ý muốn.

2. Một số thao tác sửa chữa bảng biểu

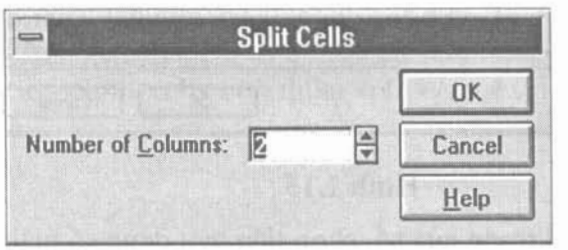
Trong một số trường hợp cần tạo nên các bảng biểu phức tạp theo yêu cầu ví dụ chia một số ô thành các ô nhỏ hơn hoặc nối thông một số ô với nhau. Ta có thể làm việc này thông qua các chức năng có trong thực đơn Table.

Hình 1.12a là bảng biểu ban đầu. Để có hình 1.12b ta bôi đen hai ô 1 và 2 sau đó chọn chức năng **Table - Merge cells.** Để có hình 1.12c ta bôi đen các ô 3,4,5 trong hình 1.12b sau đó chọn **Table - Split cells** hộp thoại split cells xuất hiện (hình 1.13). Mục chọn **Number of Columns** cho phép chia các ô đã bôi đen thành số ô tuỳ ý . Bấm vào mũi tên lên hoặc xuống để chọn số ô mà ta định chia ra , cụ thể ở đây là chia 3.



a b c

Hình 1.12



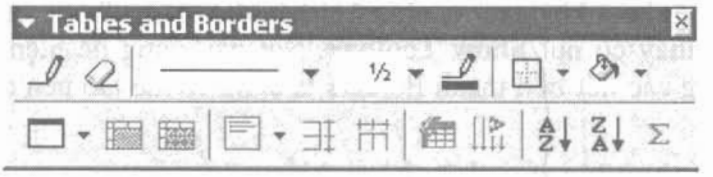
Hình 1.13

Với hệ soạn thảo Word 2000 trên thanh công cụ có thể thấy một nút vẽ bảng tên nút này là Tables and Borders

Bấm đơn để chọn nút này trên màn hình xuất hiện một thanh công cụ vẽ bảng (hình 1.14) và chuột có hình dạng bút chì, dùng bút chì chúng ta có thể vẽ một khung đồng thời vẽ thêm các nét ngang dọc nghĩa là thêm vào trong bảng các cột hoặc hàng.

Trường hợp vẽ sai bấm đơn vào viên tẩy chuột sẽ biến thành viên tẩy. Bấm rê chuột trên một nét vẽ nào đó chúng ta sẽ xoá được nét vẽ đó.

Chú ý: Những nét vẽ tạo nên đường bao của bảng khi xoá sẽ để lại nét mờ, khi in ra sẽ không có.

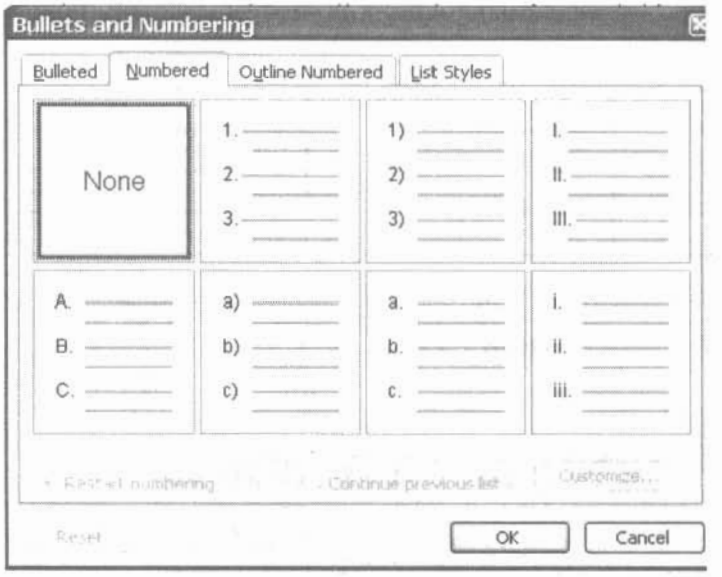


Hình 1.14

3. Đánh số thứ tự trong một cột

Giả sử cần đánh số thứ tự từ 1 đến 50 trong 50 ô của một cột nào đó

- Bôi đen số ô muốn đánh số thứ tự
- Chọn Format - chọn Bullets and Numering xuất hiện cửa sổ hình 1.15



Hình 1.15

- Chọn Numbered trong cửa sổ, chọn tiếp một dạng số mà ta muốn điền vào trong cột, ví dụ dạng đầu bên phải các số sẽ có dấu chấm. Để bỏ dấu chấm đó hãy chọn Cusstomize ta có cửa sổ hình 1.16.

Hình 1.16

Dưới mục Number Format máy để ngầm định một số và dấu chấm, hãy xoá dấu chấm đó đi rồi chọn OK.

5. Tạo bảng bằng bút chì

Sử dụng nút Tables and Borders hoặc chọn Table – Draw Table trên thực đơn ta có một thanh công cụ vẽ bảng (hình 1.14)

Lúc này chuột biến thành một chiếc bút chì, bằng cách bấm rê chuột ta có thể vẽ nên một bảng tuỳ ý, nếu nét vẽ sai có thể dùng chiếc tẩy trên thanh công cụ này để tẩy nét vẽ đi, cách thức tiến hành là:

Bấm đơn chuột vào viên tẩy, chuột sẽ có hình dạng tẩy, bấm rê chuột trên nét vẽ sai rồi buông tay ra nét vẽ sẽ biến mất.

V. CÁC THAO TÁC THƯỜNG GẶP

1. Viết chữ cái đầu mỗi đoạn

Chữ cái đầu tiên của một đoạn có thể viết to hơn các chữ khác, thậm chí có thể viết trên hai, ba dòng.

Ví dụ :

Bông lau trắng giữa rừng xanh
Mong manh trước gió , mong manh trước đời
Mùa lá đổ, mùa mưa rơi

Phất phơ lau trắng ven đồi sương buông

Các thao tác:

* Đặt con trỏ màn hình vào đầu dòng muốn điều chỉnh chữ

* Chọn chức năng **Format - Drop Cap ...** hộp thoại xuất hiện như hình 1.17.

Các khả năng lựa chọn :

None: chữ viết bình thường không viết to

Dropped: viết chữ to nằm gọn trong khuôn khổ văn bản

In Margin: chữ viết to nằm ngoài lề văn bản

Font: chọn phông chữ cho chữ viết to

Lines to Drop: chữ viết trên bao nhiêu dòng

Hình 1.17

Distance from Text: khoảng cách giữa chữ và văn bản

2. Chia trang hoặc đoạn văn bản thành nhiều cột

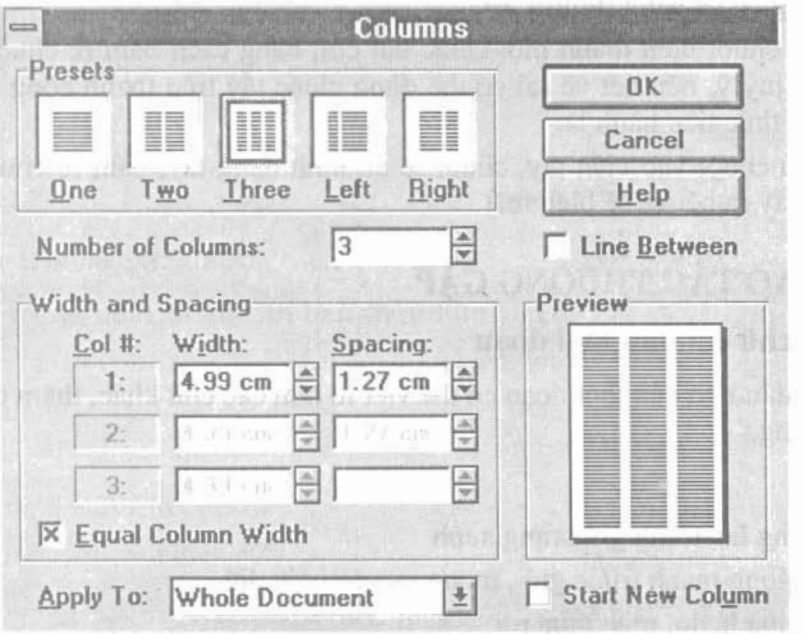
Chọn **Format - Columns** ... Hộp thoại xuất hiện như trong hình 1.18 các mục chọn:

One: trang giấy chỉ có 1 cột

Two: trang giấy chia thành 2 cột

Three: trang giấy chia thành 3 cột

Left: trang giấy chia thành 2 cột , cột bên trái có kích thước nhỏ hơn cột phải



Hình 1.18

Right : trang giấy chia thành 2 cột , cột bên trái có kích thước lớn hơn cột phải

Numbers of Columns : số lượng cột cần có

Width and Spacing :

Width : độ rộng của một cột

Spacing : khoảng cách giữa các cột

Cần chú ý rằng sau khi chọn kích thước giấy và lề trang giấy thì MS-WORD sẽ tự động xác định độ rộng của các cột tuỳ thuộc vào số lượng cột và khoảng cách giữa các cột mà ta đã chọn.

3. Tạo thuật ngữ viết tắt

Nhiều thuật ngữ đặc biệt là tiếng Latinh dùng trong y học hoặc những tên riêng chúng ta hay gặp khi soạn thảo có thể viết tắt theo phương pháp :

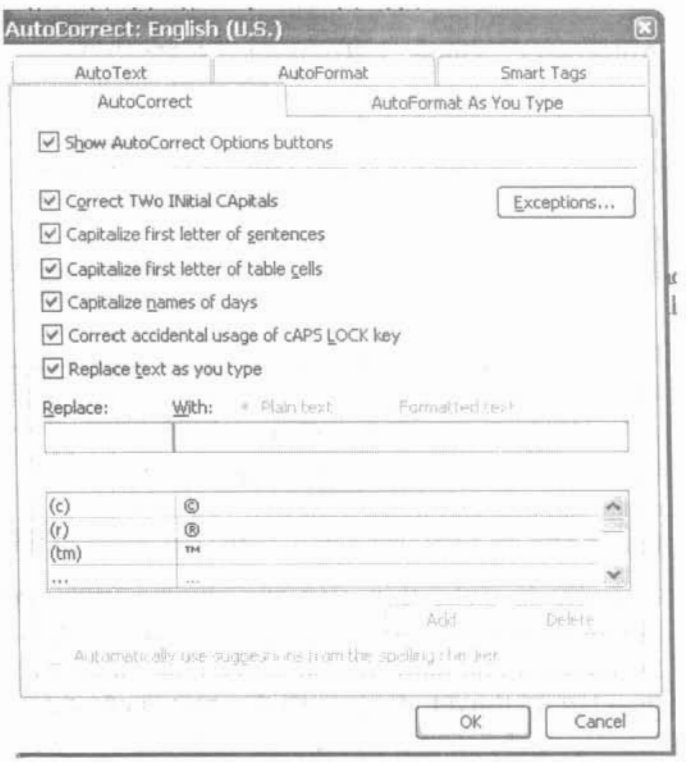
Chọn **Tools - AutoCorrect**. Hộp thoại AutoCorrect xuất hiện như hình 1.19. Giả sử chúng ta muốn viết tắt dòng chữ "Cộng hoà xã hội chủ nghĩa Việt Nam" bằng cách bấm hai chữ cái vn.

Dưới mục chọn Replace ta viết chữ "vn"

Dưới mục chọn With ta viết dòng chữ "Cộng hoà xã hội chủ nghĩa Việt Nam" sau đó bấm chuột vào nút Add rồi chọn OK.

Trong văn bản từ nay trở đi khi ta gõ vn và bấm phím khoảng cách thì toàn bộ dòng chữ Cộng hoà xã hội chủ nghĩa Việt Nam sẽ xuất hiện thay cho chữ vn.

Các từ viết tắt tạo ra như trên chỉ có thể viết được một dòng còn muốn tạo ra nhiều dòng thì phải dùng phương pháp khác.



Hình 1.19

Cần phải chú ý rằng khi viết trong khung With các ký tự tiếng Việt có thể sẽ không hiện bình thường, hãy cứ gõ đúng sau này chữ sẽ hiện bình thường.

5. Tạo đoạn văn bản mẫu

Nếu hàng này chúng ta phải thường xuyên gửi công văn đi các nơi thì phần tiêu đề của công văn có thể tạo sẵn và lưu trong máy sau đó chỉ việc gọi ra và điền nội dung vào.

Giả sử cần tạo ra một tiêu đề mẫu sau đây:

Cộng hoà xã hội chủ nghĩa Việt Nam

Độc lập - Tự do - Hạnh phúc

Công văn

Hà Nội, ngày ..... tháng ..... năm 199...

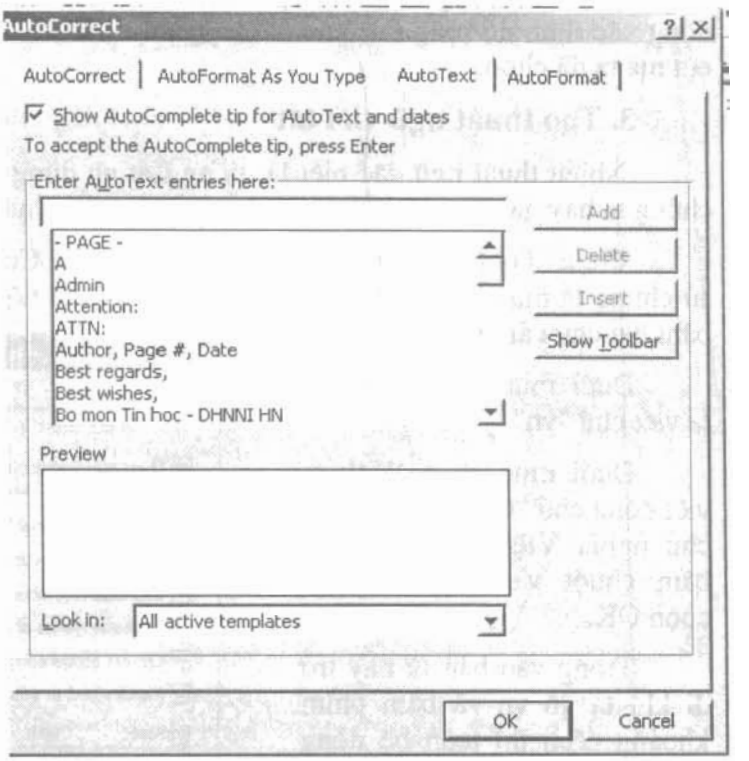
Các bước thực hiện:

* Viết đoạn tiêu đề công văn

* Trình bày kiểu chữ theo ý muốn.

* Bôi đen đoạn tiêu đề.

* Chon chức năng Tools - Autocorect - AutoText ... xuất hiện cửa sổ hình 1.20.



Hình 1.20

* Trong hộp thoại Autocorect (hình 1.20) dưới mục Enter Autotext entries here ta viết một tên cho đoạn văn bản này. Tên cần chọn sao cho dễ nhớ và chỉ được viết bằng tiếng Việt không dấu. Tiếp đó bấm đơn vào nút Add.

Khi cần điền đoạn tiêu đề trên vào trang văn bản ta thực hiện các bước:

Chọn Insert - AutoText, tìm trong khung lựa chọn tên của đoạn văn bản cần thiết, bấm đơn vào tên đó cho nó nằm trong nền xanh tiếp đó chọn tiếp nút Insert .

Do hạn chế thời gian của chương trình học tập trong tài liệu này chỉ giới thiệu những gì mà sinh viên có thể tiếp thu trong khoảng 6 tiết học. MS-WORD 2000 là một hệ soạn thảo đa năng với rất nhiều chức năng phong phú muốn khai thác hết cần có thời gian học lý thuyết dài hơn và đặc biệt là cần được thao tác trên máy nhiều hơn.

5. Viết số mũ và chỉ số

* Viết số mũ : ví dụ x^3

Đầu tiên viết chữ x sau đó bấm tổ hợp phím Ctrl+ Shift + =, con trỏ lúc này sẽ chuyển lên vị trí cao để ta gõ số 3 , gõ lại một lần nữ tổ hợp phím trên con trỏ sẽ trở lại bình thường.

* Viết chỉ số : ví dụ x_1

Gõ chữ x sau đó bấm tổ hợp phím Ctrl + =, con trỏ dịch xuống vị trí thấp ta bấm tiếp số 1 rồi bấm lại tổ hợp phím trên một lần nữa.

6. Chèn vào văn bản một ký tự đặc biệt

Chọn Insert - Symbol ta sẽ có một bảng các ký tự đặc biệt, các ký tự này thường rất nhỏ khó nhận dạng, muốn nhìn rõ bấm đơn vào ký tự đó. Muốn chèn ký tự này vào văn bản bấm kép vào ký tự hoặc bám vào nút Insert ở phía dưới . Xong việc bấm Close.

VI. TỰ ĐỘNG HOÁ QUÁ TRÌNH SOẠN THẢO

1. Tạo dòng Tab

Khi bấm phím Tab trên bàn phím, con trỏ sẽ tự động dịch chuyển sang phải một đoạn, Word ngầm định đoạn này là 1,27 cm (1/2 Inch). Có thể thiết kế lại Tab để khi dịch chuyển con trỏ sẽ để lại trên dòng các dấu chấm, các nét gạch ngang. Giả sử ta cần tạo ra dòng văn bản sau:

Họ và Tên: ..... Ngày sinh: ..... Dân tộc: .....
8 cm 5 cm 3 cm

Chữ Họ và tên viết cách mép trái 1 cm. Phần dành cho Họ và tên là 8 cm, phần đành cho ngày sinh là 5 cm và phần dành cho dân tộc là 3 cm. Cách thức tạo như sau:

Chọn Format - Tab ta có cửa sổ (hình 1.21). Trong các khung khai báo ta tiến hành khai báo như sau:

Tab Stop Position : gõ 1
Alignment : chọn Left
Leader : chọn None
Chọn Set

Đây là khai báo vị trí đầu tiên của Tab cách lề trái 1 cm.

Tiếp đó lặp lại các thao tác
Tab Stop Position : gõ 8
Alignment : chọn Left
Leader : chọn số 2
Chọn Set
Tab Stop Position : gõ 5
Alignment : chọn Left
Leader : chọn số 2
Chọn Set
Tab Stop Position : gõ 3
Alignment : chọn Left
Leader : chọn số 2
Chọn Set

Hình 1.21

Chon OK để trở về của sổ soạn thảo, khi đó trên dòng thước kẻ (Ruler) xuất hiện một số ký hiệu L thể hiện vị trí Tab.

Dòng Tab vừa tạo ra sẽ có tác dụng cho đến khi ta tạo ra dòng Tab mới.

2. Tạo các STYLE

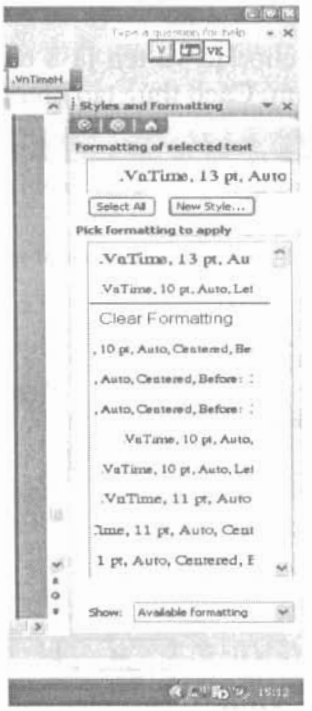
Style được hiểu là tập hợp những thao tác định dạng cho một đoạn văn bản do người sử dụng lựa chọn, chúng được lưu trữ dưới một tên gọi và có thể sử dụng bất kỳ lúc nào. Khi một Style được gọi toàn bộ phần văn bản viết sau đó sẽ được định dạng theo Style này. Phần định dạng đưa vào một Style có thể là kiểu chữ, dạng chữ, kích thước, màu sắc chữ, khoảng cách giữa các dòng, giữa các cột, giữa các đoạn ....

Giả sử cần tạo ra Style có tên là St1 dùng cho các đề mục a, b, … trong giáo trình này , trong St1 sẽ đưa vào các định dạng:

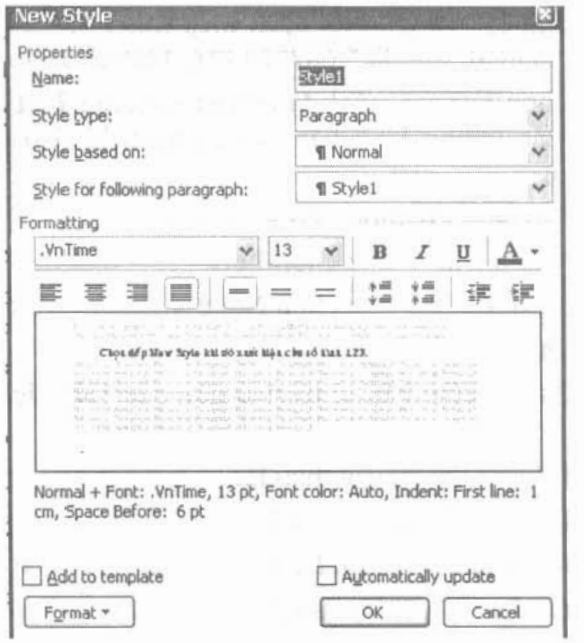
Kiểu chữ : VnTime , kích thước :13, chữ nghiêng, cách đoạn trên và cách đoạn dưới 3 Pt. Phương pháp tiến hành như sau:

a. Tạo Style mới:

Chọn Format – Style and Formating khi đó phần bên phải màn hình xuất hiện một cửa sổ (hình 1.22).



Hình 1.22



Hình 1.23

Chọn tiếp New Style khi đó xuất hiện cửa sổ hình 1.23.

Trong mục **Name** (tên của Style mới) ta gõ St1.

Mục **Style Type** có hai lựa chọn:

Character để định dạng chữ, **Paragraph** để định dạng đoạn văn bản, ngầm định là Paragraph.

Giả sử ta chọn Paragraph, sau đó chọn nút Format khi đó sẽ xuất hiện hộp thoại bao gồm:

Font : định dạng chữ

Paragraph : định dạng đoạn

Tabs : định dạng bước nhảy khi bấm phím Tab

Border : định dạng đường bao (bảng biểu)

Language ; chọn ngôn ngữ

Frame : định dạng khung

Numbering : định dạng kiểu số

Chọn một trong các chức năng của hộp thoại ví dụ chọn Font ta sẽ quay về cửa sổ quen thuộc để làm các thao tác định dạng, cụ thể ta sẽ chọn: Kiểu chữ: VnTime, kích thước :13, chữ nghiêng.

Chọn tiếp mục Paragraph ta sẽ có cửa sổ Paragraph, trong các khung Before và After chọn 3 pt.

Sau khi đã lựa chọn xong chọn OK để quay về các bước trước đó, cuối cùng chọn Close để trở về màn hình văn bản hiện thời. Nếu muốn ứng dụng ngay Style vừa tạo ra thì chọn Apply.

Tên của Style vừa tạo ra được để trong thanh định dạng. Nếu muốn ứng dụng Style nào ta chỉ việc kích chuột vào mũi tên bên phải nút Style tên các Style sẽ hiện lên trong hộp lựa chọn.

3. Tạo Macro

Thuật ngữ Macro thường được dịch sang tiếng Việt là "vĩ lệnh" hay lệnh tổ hợp, nó giống như một máy ghi, ghi lại toàn bộ các thao tác mà người sử dụng đã tiến hành. Khi gọi tên Macro các thao tác này sẽ được lặp lại từ đầu cho đến thao tác cuối cùng.

Macro có thể được lưu trữ trên thanh công cụ, trên bàn phím (với các phiên bản trước Window2000 Macro còn có thể lưu trong Menu.)

3.1. Tạo Macro trên ToolsBar

Các bước tiến hành:

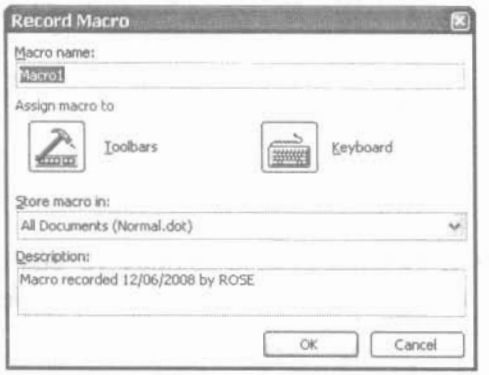
a. Chọn Tools - Macro - Record new macro, xuất hiện cửa sổ (hình 1.24)

b. Trong mục Macro Name ta ghi một tên nào đó ví dụ M1

c. Trong mục Assign Macro to (đặt macro vào) Chọn Toolbars có cửa sổ tiếp theo (hình 1.25), cách chọn này có nghĩa là Macro sẽ lưu thành một nút trên thanh công cụ, nếu chọn KeyBoard thì Macro được lưu trên bàn phím.

d. Trong hình 1.25 bấm đơn vào phiếu chọn Command, dưới mục Commands bấm và di cả dòng chữ Normal.NewMacros.Macro1 lên thanh công cụ sẽ tạo nên một nút mới, bên trong nút này chưa có biểu tượng mà chỉ có dòng chữ đã nói. Để thu bé biểu tượng chọn tiếp Modify Selection sẽ xuất hiện cửa sổ hình 1.26.

Trong hình 1.26 bên phải mục Name trong hộp thoại hãy xoá bớt số ký tự đã có và điền vào đó tên mà ta lựa chọn. Cần lưu ý rằng không thể xoá bỏ tất cả mọi ký tự mà phải điền vào ít nhất là một ký tự.



Hình 1.24

Hình 1.25

Mục Change Button Image cho phép lựa chọn một hình vẽ cho biểu tượng macro, chỉ cần bám đơn vào biểu tượng là nó sẽ tự động được đưa vào bên trong nút mà ta đã đặt trên thanh công cụ.

Chọn Close để quay về cửa sổ soạn thảo, lúc này trên màn hình xuất hiện một thanh công cụ gồm hai nút:

Stop : dùng để ngừng ghi các thao tác vào Macro

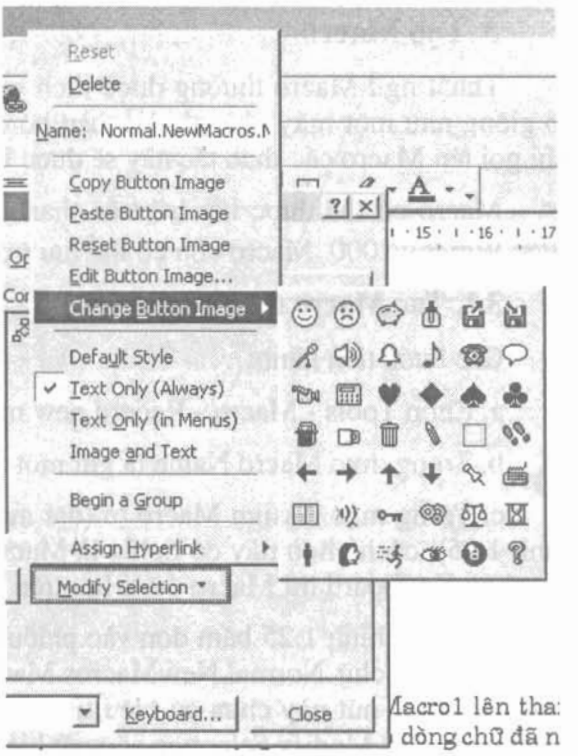
Pause : dùng để tạm ngừng ghi Macro.

Con trỏ chuột trên màn hình xuất hiện thêm một biểu tượng giống như băng cát xét. Kể từ thời điểm này mọi thao tác trên bàn phím sẽ được ghi lại và được gửi vào biểu tượng vừa tạo ra trên thanh công cụ.

Kết thúc công việc tạo Macro bấm vào nút Stop.

Chú ý : Khi thực hiện các thao tác tạo Macro, ta không thể bôi đen văn bản bằng thiết bị chuột mà phải dùng phím Shift và các mũi tên dịch chuyển.

* Để gọi Macro ta chỉ việc bấm vào biểu tượng đã tạo ra trên thanh công cụ.



Hình 1.26

* Để xoá một Macro đã có trên thanh công cụ ta đè phím Alt đồng thời bấm rê kéo biểu tượng Macro vào vùng soạn thảo, biểu tượng sẽ biến mất.

* Để xoá tên Macro chọn: Tools - Macro - chọn tên Macro rồi chọn Delete

3.2. Tạo Macro trên bàn phím (KeyBoard)

Các bước tiến hành:

a. Lặp lại các bước a,b trong mục 3.1 ở trên

b. Chọn KeyBoard sẽ xuất hiện cửa sổ mới (hình 1.27)

Trong mục chọn Press New Shortcut Key ta bấm một tổ hợp phím, ví dụ Ctrl - Alt - A

c. Chọn Assign, tổ hợp phím trên được ghi vào vùng Current Key

d. Chọn Close để trở về màn hình soạn thảo

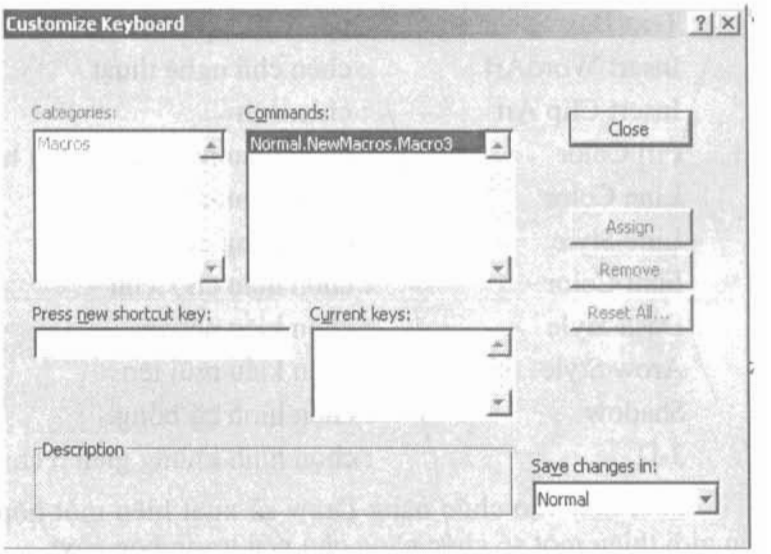
Các thao tác tiếp theo sẽ lặp lại như mục 1

e. Gọi Macro

Để gọi một Macro tiến hành các thao tác sau:

* Đặt con trỏ ở đầu một dòng trống

* Chọn Tools - Macro - Macro xuất hiện cửa sổ mới



Hình 1.27

* Chọn tên Macro trong danh mục sau đó chọn Run

Cũng có thể cho Macro thực hiện bằng cách bấm tổ hợp phím mà bạn đã chọn khi tạo macro.

VII. ĐỒ HOẠ

1. Các thao tác vẽ cơ bản

Đồ hoạ là phần công cụ phụ trợ trong Winword, ở đây chỉ bao gồm những chức năng cơ bản, nếu muốn sử dụng công cụ đa năng hơn nhằm mục đích trình chiếu ta có thể dùng PowerPoint là một ứng dụng độc lập với Winword và cũng được cài đặt trong Windows.

Chọn nút Drawing trên thanh công cụ ta sẽ có thêm một thanh công cụ mới, thanh này thường được bố trí dưới đáy màn hình (hình 1.28a). Nếu máy cài đặt bộ Office 97 hoặc office 4.3 thì thanh công cụ vẽ có thể khác đi đôi chút vì đó là các phiên bản cũ.

Hình 1.28a

Thanh công cụ đồ hoạ bao gồm các nút có tên như sau (thứ tự từ trái sang phải):

Draw : kẻ lưới, quay hình, chọn thứ tự hình ....

Select Object : chọn đối tượng vẽ

Free Rotace	: quay hình tự do
AutoShapes	: chọn các hình mẫu có sẵn
Line	: vẽ đường thẳng
Arrow	: vẽ mũi tên
Rectangle	: vẽ hình vuông hoặc chữ nhật
Oval	: vẽ hình tròn hoặc Ellipse
Text Box	: tạo một hộp chứa văn bản
Insert WordArt	: chèn chữ nghệ thuật
Insert Clip Art	: chèn ảnh
Fill Color	: chọn màu nền cho những hình khép kín
Line Color	: chọn màu nét vẽ
Line Style	: chọn dạng nét vẽ
Font Color	: chọn màu cho chữ
Dash Style	: chọn kiểu nét vẽ
Arow Style	: chọn kiểu mũi tên
Shadow	: chọn hình có bóng
3-D	: chọn hình không gian 3 chiều

Bấm đơn vào chức năng Draw sẽ xuất hiện một hộp thoại (hình 1.28b), dưới đây xin giới thiệu một số chức năng chủ yếu trong hộp này:

Set AutoShape Defaueflts : quy ước hình vẽ ngầm định

Change Autoshape: thay đổi các hình vẽ có sẵn trong thư viện đồ hoạ

Edit Points : thay đổi hoặc sửa chữa các điểm trên hình

Text Wrapping: cách thức bố trí văn bản xen kẽ hình vẽ

Rotate or Flip: quay hình

Align or Distribute: dóng lề các hình

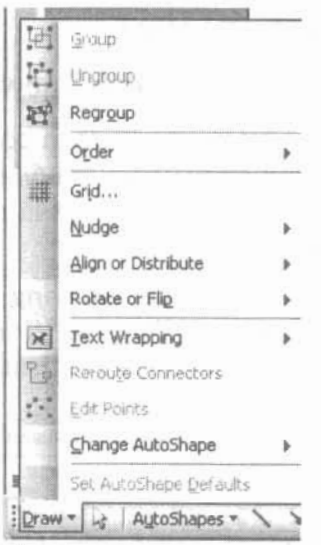
Grid: kẻ lưới trên hình vẽ

Order: thứ tự bố trí hình trên hay dưới

Regroup: tái kết nối các hình

Ungroup: không kết nối các hình nữa

Group: kết nối các hình thành một hình



Hình 1.28b

Trong mục Order chúng ta có thể tìm được các chức năng sau (có thể một vài chức năng đã bị xoá):

Bring to Front : chuyển một hình vẽ ở dưới lên phía trên nhiều hình khác

Send to Back: chuyển hình vẽ ở trên xuống dưới nhiều hình khác

Bring Forward: đưa một hình lên trên một hình khác

Send BackWard: đưa một hình xuống dưới một hình khác

Bring in Front of Text : đưa hình che lên trên văn bản

Send Behind Text: cho văn bản hiện lên trên hình

Để vẽ hình chúng ta cần thực hiện các bước sau đây;

1. Chọn một trong năm dạng nét vẽ (Line, Rectangle, Ellipse....) bằng cách bấm đơn vào biểu tượng nét vẽ đó, chuyển chuột vào vùng soạn thảo, chuột sẽ biến thành hình dấu "+".

2. Bấm di chuột để tạo hình có kích thước mong muốn.

3. Đánh dấu hình: Đưa chuột đến gần hình chuột sẽ xuất hiện thêm biểu tượng hai mũi tên vuông góc. Bấm đơn để đánh dấu hình (xuất hiện các khối vuông đen bao quanh hình)

4. Tô màu:

* Nếu là nét vẽ không khép kín thì chỉ có thể tô màu nét vẽ,

Chọn Line Color - chọn một màu nào đó

* Nếu là nét vẽ khép kín (hình vuông , tròn, tam giác....) việc chọn màu nét vẽ như mục 3, để chọn màu nền bấm đơn vào nút Fill Color - rồi chọn một màu ưng ý.

5. Chọn dạng nét vẽ : Đánh dấu hình rồi chọn nút Line Style - sau đó chọn một nét thích hợp.

6. Muốn đưa một lời chú thích và mũi tên chỉ vào một vị trí nào đó:

- Chọn AutoShapes
- Chọn Callout, chọn một dạng hình có sẵn
- Đặt chuột (dấu "+") vào vị trí cần chỉ mũi tên
- Bấm di chuột sẽ tạo ra một hình để ghi chú thích bên trong

Một số điều cần lưu ý :

*** Chọn nhiều hình cùng một lúc**

Đè phím Shift rồi đánh dấu lần lượt từng hình

*** Chuyển vị trí hình**

Khi vẽ hình, hình vẽ sau sẽ che lên trên hình vẽ trước. Muốn thấy hình vẽ trước ta thực hiện :

- Đánh dấu hình vẽ sau
- Chọn Draw - Order
- Chọn nút Send to Back

Ngược lại hình vẽ sau có thể đưa lên trước bằng cách đánh dấu hình rồi chọn nút Bring to Front

* Nếu có hai hình nằm theo hàng ngang và cả hai hình đang được đánh dấu ta có thể:

quay hình đi 90° (nút Rotate Right), chuyển hình trái sang phải (Flip Horizontal)

* Nếu có hai hình nằm theo hàng dọc và cả hai hình đang được đánh dấu ta có thể: quay hình đi 90° (nút Rotate Right), chuyển hình trên xuống dưới (Flip Vertical)

*** Viết chữ trong hình :**

- Chọn nút Text Box rồi vẽ trong hình một khung, trong khung sẽ có con trỏ để viết văn bản

- Nếu không có Text Box ta có thể viết trực tiếp trong hình, chữ viết ra không nhìn thấy vì nằm bên dưới hình, muốn chữ hiện lên:

- Đánh dấu hình

- Chọn nút : Send Behind text

Chú ý : *Để có thể đưa con trỏ vào trong hình vẽ, trước khi vẽ cần đưa con trỏ xuống đáy màn hình. Việc dịch chuyển con trỏ sang ngang phải dùng phím Space (phím khoảng cách)*

*** Dóng vị trí (canh lề) các hình:**

-Vẽ hai hình ở vị trí bất kỳ trên màn hình

- đánh dấu cả hai hình.

- chọn Align or Distribute xuất hiện hộp thoại (hình 1.29).

Horizontal : (theo chiều ngang)

Align Left: fóng thẳng theo cạnh trái của hình

Align Center: đặt hình vào giữa theo chiều ngang

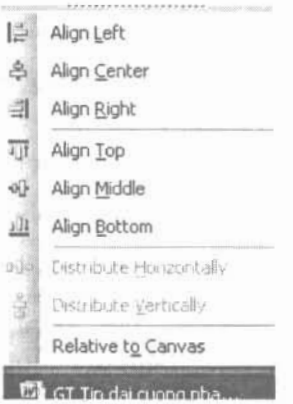
Align Right: dóng thẳng theo cạnh phải của hình

Align Top: dóng thẳng theo mép trên của hình

Align Middle: dóng thẳng theo đường xuyên tâm theo chiều ngang

Align Bottom: dóng thẳng theo mép dưới của hình

Distribute Horizontalty: đảo vị trí hình đang được chọn theo chiều ngang (nếu hai hình cùng nằm ngang)



Hình 1.29

Distribute Vertically: đảo vị trí hình đang được chọn theo chiều dọc (nếu hai hình cùng nằm theo chiều dọc).

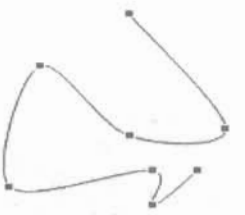
*** Bẻ gẫy nét vẽ :**

- Chọn AutoShapes - Lines - freedom vẽ 1 nét bất kỳ bao gồm một số đoạn thẳng và đoạn cong (hình 1.30), đánh dấu nét vẽ

- Chọn Draw - Edit Poitns nét vẽ sẽ được đánh dấu tại các điểm giao nhau (hình 1.31) và chuột biến thành một hình vuông nhỏ rỗng bên trong.

Trên nét vẽ các đoạn thẳng sẽ được đánh dấu ở hai đầu, còn các đoạn cong sẽ được đánh dấu liên tục bởi các ký hiệu đánh dấu. Đưa chuột vào một vị trí bất kỳ trên nét vẽ bấm rê chuột ta sẽ uốn được nét vẽ theo ý muốn.

Hình 1.30



Hình 1.31

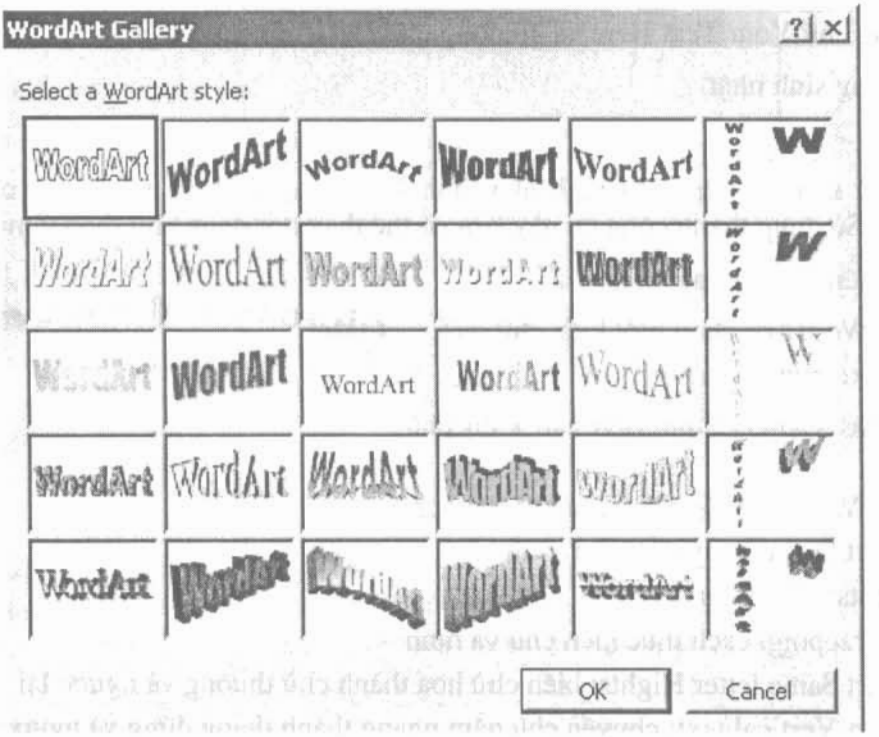
VIII. CHỮ NGHỆ THUẬT

Chữ nghệ thuật được hiểu là các kiểu chữ phi tiêu chuẩn đã được thiết kế sẵn trong Word. Chữ nghệ thuật được sử dụng để viết các khẩu hiệu, trang trí trên các hình vẽ v.v... Thao tác viết chữ nghệ thuật được thực hiện qua các bước sau:

Bấm đơn vào nút đồ hoạ Drawing trên thanh công cụ chuẩn lúc này trên màn hình xuất hiện thanh công cụ đồ hoạ.

Bấm đơn vào nút Insert WordArt xuất hiện cửa sổ WordArt Gallery (hình 1.32).

Chọn một kiểu chữ nghệ thuật mà bạn muốn rồi bấm OK sẽ xuất hiện cửa sổ Edit WordArt Text (hình 1.33).



Hình 1.32

Hình 1.33

Font chữ hiện thời là font tiếng Anh. Để chọn Font tiếng Việt trong mục Font kích chuột vào mũi tên bên phải sau đó chọn một Font tiếng Việt, ví dụ chọn VnAristote.

Trong mục Size có thể chọn kích thước chữ, ngoài ra bạn có thể chọn kiểu chữ đậm (B), nghiêng (I).

Xoá dòng chữ Your Text Here và gõ dòng chữ cần trình bày ví dụ :

Chúc mừng sinh nhật

Chọn OK

Dòng chữ sẽ hiên trên màn hình như hình 1.34 đồng thời có một thanh công cụ hiện kèm theo. Sử dụng thanh công cụ này bạn có thể thay đổi dòng chữ theo ý muốn.

Các nút trên thanh công cụ này là:

Hình 1.34

* Insert WordArt: chọn một kiểu chữ mới
* Edit Text: sửa lại chữ cũ
* WordArt Gallery: thư viện các kiểu chữ nghệ thuật
* Format WordArt: định dạng nét và màu chữ
* WordArt Shape: thay đổi kiểu trình bày
* Free Rotate: quay tự do
* Text Wrapping: cách thức hiện chữ và hình
* WordArt Same letter Hights: biến chữ hoa thành chữ thường và ngược lại
* WordArt Vertical text: chuyển chữ nằm ngang thành thẳng đứng và ngược lại
* WordArt Agliment: canh lề chữ
* WordArt Character Spacing: chọn chữ gầy hay béo

Hình 1.35 cho ta một dạng của chữ nghệ thuật, muốn có chữ kiểu con dấu, ta phải gõ vào vùng your text here ba dòng khác nhau. Ví dụ:

Trên cùng là dòng
"Trường Đại học Chu Văn An",
tiếp đó là dòng "Bộ môn Tin học"
Dưới cùng là dòng
" Khoa Công nghệ thông tin "

Hình 1.35

Chú ý :

* Việc chọn Font chữ hoặc chọn màu chỉ có thể thực hiện trong cửa sổ của Word Art mà không thể chọn theo cách thông thường.

* Muốn sửa lại một WordArt đã có, bấm kép vào vùng chữ đã được tạo ra ta sẽ trở lại cửa sổ WordArt.

* Với một WordArt đã có ta có thể mở rộng hoặc thu hẹp kích thước bằng cách bấm đơn vào vị trí bất kỳ thuộc Word Art đó, lúc này toàn bộ Word Art sẽ nằm trong một hình chữ nhật. Các cạnh của hình chữ nhật được đánh dấu bởi các hình tròn rỗng. Đặt chuột vào một trong các hình đó, bấm rê chuột sẽ thay đổi được kích thước của WordArt.

IX. TRỘN TÀI LIỆU TẠO THÀNH VĂN BẢN MỚI

Nội dung :

Trộn dữ liệu trong một bảng biểu đã có sẵn (đã lưu trong đĩa cứng với một tên nào đó) với một văn bản mẫu để tạo thành một văn bản mới. Ví dụ đã có sẵn một mẫu giấy mời và một danh sách khách mời ta sẽ trộn chúng với nhau để có một tập giấy mời hoàn chỉnh.

* Các bước tiến hành:

- Tạo sẵn mẫu giấy mời và ghi vào đĩa với tên là GM.
- Tạo một bảng biểu chứa danh sách những người sẽ mời
- Trộn lẫn hai văn bản này thành một giấy mời hoàn chỉnh

1. Tạo mẫu giấy mời theo mẫu dưới đây

Cộng hoà xã hội chủ nghĩa Việt Nam

Độc lập -Tự do-Hạnh phúc

Giấy mời

Trân trọng kính mời Ông/Bà:
Đến dự :
Thời gian :
Địa điểm :

Rất hân hạnh được đón tiếp

Ngày tháng năm 1996

* Ghi giấy mời vào đĩa với tên GM

2. Tạo danh sách khách mời

* Tạo một bảng chứa danh sách khách mời (bảng này không nằm chung với giấy mời), ví dụ bảng có bảng 4 cột 6 dòng theo mẫu dưới đây:

Hoten	noidung	thoigian	diadiem
Nguyễn Văn Tâm	Dự họp tổng kết	8h30	Hội trường 204
Trần Thanh Bình	Dự liên hoan chia tay	11h30	Nhà hàng Hương sen
Cao Thi Vân	Học nghị quyết	7h30	Phòng 12 - A1
Đào Hoàng Lâm	Họp thi đua	7h30	Phòng hội thảo
Phạm Minh Hồng	Dự cưới	10h30	Nhà riêng

Lưu ý :

Dòng đầu tiên của bảng biểu là Tiêu đề do đó không viết tiếng việt có dấu, không được viết các chữ cách nhau - Không trang trí đường viền của bảng biểu

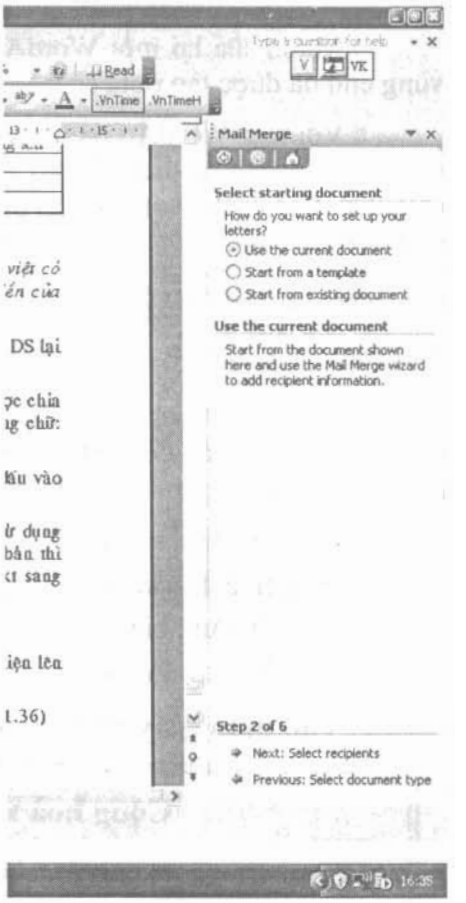
* Ghi bảng biểu này vào với tên DS (danh sách). Đóng cửa sổ DS lại cho hiện GM lên màn hình.

* Chọn Tools – Letter and Mailing – Mail Merge màn hình được chia thành hai cửa sổ (Hình 1.36). Trong cửa sổ bên phải dưới đáy có dòng chữ: Step 1 of 6 nghĩa là việc trộn tài liệu sẽ qua 6 bước.

- Step 1 of 6 (bước 1): chọn loại văn bản đang sử dụng, đánh dấu vào vòng tròn bên trái chữ Letters. Bấm vào Next để sang bước 2.

- Step 2 of 6: đánh dấu vào mục Use the current document (sử dụng văn bản đang có trên màn hình). Nếu trên màn hình chưa có văn bản thì chọn khả năng thứ ba (start from existing document).
Bấm vào Next sang bước 3.

Step 3 of 6: bước này cần chọn một bảng dữ liệu để trộn với văn bản hiện thời, có ba khả năng lựa chọn;



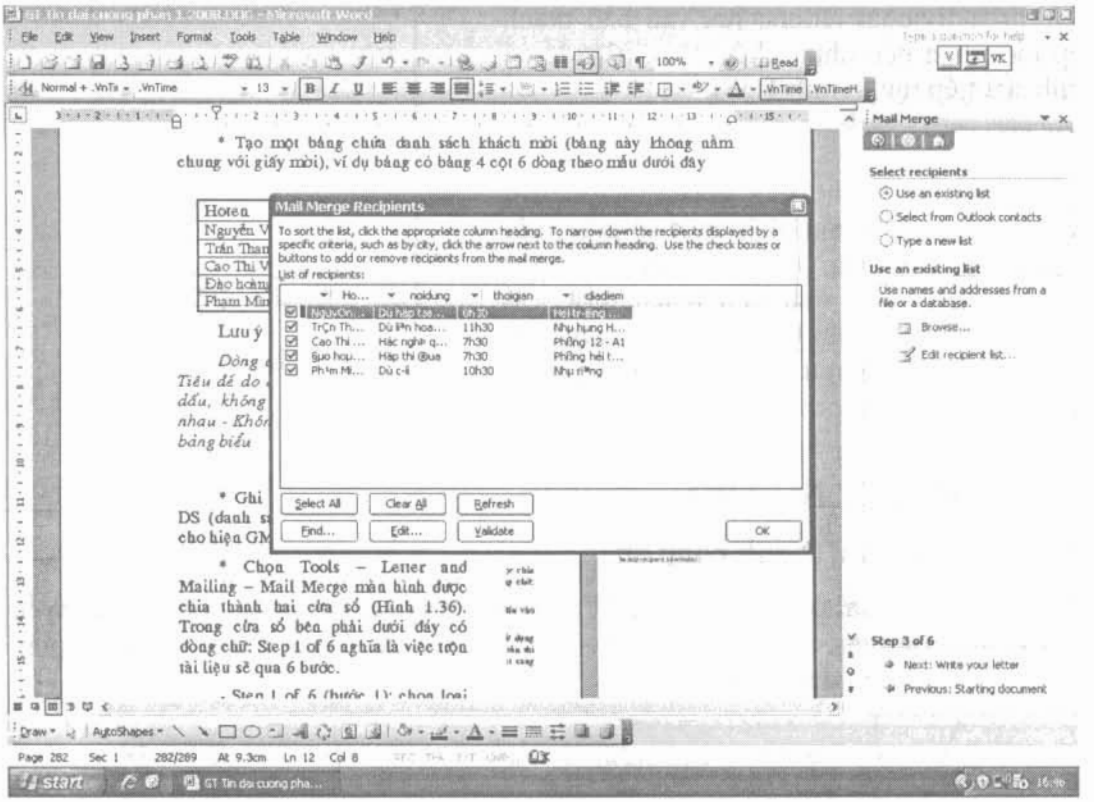
Hình 1.36

- Use an existing list: chọn bảng dữ liệu đã lưu trong đĩa.

- Select from Outlook contacts: bảng dữ liệu được nhập từ nguồn khác

- Type a new list: tạo một bảng dữ liệu mới.

Chọn khả năng thứ nhất Use an existing list lúc này một cửa sổ hiện ra cho phép tìm bảng dữ liệu DS chúng ta đã lưu trong máy. Tìm thấy hãy chọn Open để mở DS bạn sẽ có màn hình mới (hình 1.37).



Hình 1.37

Muốn trộn tất cả các dòng dữ liệu của DS với GM hãy bấm vào nút Select all, lúc này bên trái các dòng xuất hiện ký hiệu đánh dấu. Bấm OK rồi chọn Next để sang bước 4.

Step 4 of 6: đây là bước điền dữ liệu từ DS vào GM, hãy bấm đơn chuột để đặt con trỏ vào bên phải mục: Trân trọng kính mời Ông/Bà sau đó chọn More items… ở cửa sổ bên phải màn hình chúng ta có cửa sổ mới (hình 1.38). Trong cửa sổ này ta chỉ thấy tiêu đề các cột trong tệp DS và ta cần đưa họ và tên khách mời vào bên phải dòng chữ "Trân trọng kính mời…. Hãy bấm đơn vào dòng Hoten rồi chọn Insert. Lúc này chúng ta chưa thấy họ tên hiện lên mà chỉ thấy một cụm ký tự lạ, hãy yên tâm và tiếp tục với các cột còn lại. Với các cụm ký tự lạ mới xuất hiện bạn có thể bôi đen rồi chọn font chữ, kiểu chữ, màu sắc … khi chọn xong hãy bấm next để sang bước 5

- Step 5 of 6: bước này cho phép quan sát văn bản sau khi trộn, trên màn hình là dữ liệu từ dòng đầu tiên của DS trộn với mẫu giấy mời GM, muốn xem các dòng tiếp theo hãy bấm vào mũi tên kép ở cửa sổ bên phải. Nếu không có gì sai sót hãy bấm next để sang bước 6.

- Step 6 of 6: bước này cho phép chọn nơi lưu trữ văn bản sau khi trộn.

- Print... in tài liệu đã trộn ra máy in (chú ý máy in phải sẵn sàng)

- Edit individual letters: lưu văn bản thành một tệp mới, bạn nên chọn chức năng này để có thể chỉnh sửa tiếp tục. (Hình 1.38).

Hình 1.38

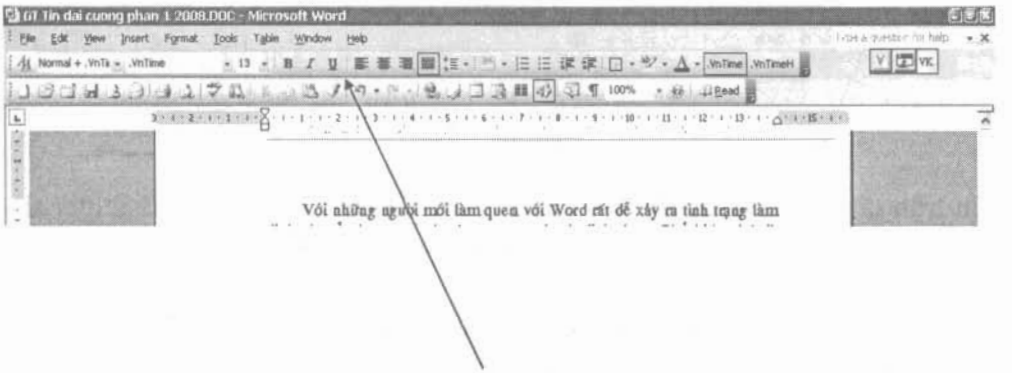
Chú ý:

Sau khi trộn có thể mỗi giấy mời sẽ nằm trên một trang giấy. Muốn ghép một số giấy mời vào một trang hãy bấm chuột để đưa con trỏ xuống dưới giấy mời đầu tiên (cách một, hai dòng) sau đó bấm hai lần phím Delete, giấy mời phía dưới sẽ được kéo lên trang hiện thời, tiếp tục với các giấy mời còn lại.

X. CÁC THAO TÁC HỖ TRỢ

1. Khôi phục các thanh công cụ

Với những người mới làm quen với Word rất dễ xảy ra tình trạng làm dịch chuyển hoặc mất thanh công cụ, thanh định dạng. Phổ biến nhất là thanh định dạng nằm phía trên, thanh công cụ nằm dưới (hình 1.39).



Hình 1.39. Vị trí bấm và di chuột

Để đưa thanh định dạng xuống dưới ta chỉ việc đưa chuột vào vị trí phân cách giữa các nút hoặc nhóm nút chuột sẽ biến thành mũi tên bốn chiều sau đó đè phím trái chuột, lúc này đường viền thanh sẽ biến thành nét đứt. Bấm và di thanh xuống phía dưới. Khi thanh định dạng nằm trùng lên thanh công cụ thì buông nút trái chuột.

2. Hiện hoặc giấu các thanh công cụ

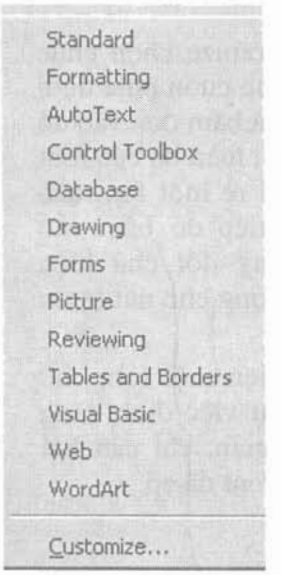
Ngầm định trên màn hình chỉ có hai thanh công cụ là thanh Standard và thanh Formating tức là thanh công cụ chuẩn và thanh định dạng. Một số nút trên thanh

Standard như nút Drawing (đồ hoạ), nút Table and Border (vẽ bảng) khi chọn lại cho hiện lên thanh công cụ tương ứng. Việc cho hiện hoặc giấu các thanh công cụ có thể thực hiện theo các bước sau đây:

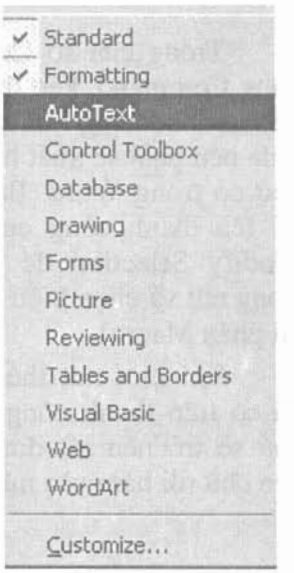
1. Khi một nút ứng với một thanh nào đó đang có màu trắng (nghĩa là nó đang được chọn) thì thanh công cụ tương ứng sẽ xuất hiện trên màn hình. Bấm đơn vào nút này thanh công cụ tương ứng sẽ biến mất và nút trở thành màu xám. Bấm tiếp vào nút thanh công cụ sẽ lại hiện ra.

2. Nếu trên màn hình không có một thanh công cụ nào, hoặc thiếu thanh công cụ cần thiết, đưa chuột vào một vị trí bất kỳ trên thanh thực đơn, bấm phím phải chuột sẽ xuất hiện một hộp thoại (hình 1.40).

Bấm đơn vào tên thanh công cụ muốn chọn thanh này sẽ xuất hiện trên màn hình, thao tác tương tự với các thanh khác.



Hình 1.40



Hình 1.41

Chú ý:

a. Nếu phía bên trái tên thanh công cụ đã có ký hiệu đánh dấu (hình 1.41) mà trên màn hình vẫn không thấy thanh công cụ đó thì cần tìm xem nó nằm đâu đó trên màn hình, có khi chỉ nhìn thấy một phần rất nhỏ của thanh công cụ, phần còn lại nằm khuất ngoài phạm vi màn hình. Chỉ cần đưa chuột vào phần nhỏ đó bấm rê nó vào trong màn hình là được.

b. Không nên cho hiện quá nhiều thanh công cụ trên màn hình vì như vậy sẽ không còn không gian để làm việc. Khi dùng xong một thanh công cụ nào nếu không cần thiết thì nên cất nó đi.

3. Thêm bớt chức năng trên thanh công cụ

Mỗi thanh công cụ của Word có một số biểu tượng (nút) thể hiện một số chức năng, chúng ta có thể tuỳ ý thêm bớt các chức năng vào các thanh này hoặc chuyển các nút từ thanh này sang thanh khác.

3.1. Xoá hoặc chuyển vị trí một nút

Đè phím Alt trên bàn phím sau đó bấm rê nút muốn xoá vào vùng soạn thảo, buông hai tay ra nút sẽ bị xoá.

Nếu chúng ta lôi nút đến một thanh công cụ khác rồi buông tay ra thì nút sẽ nằm trên thanh công cụ đó.

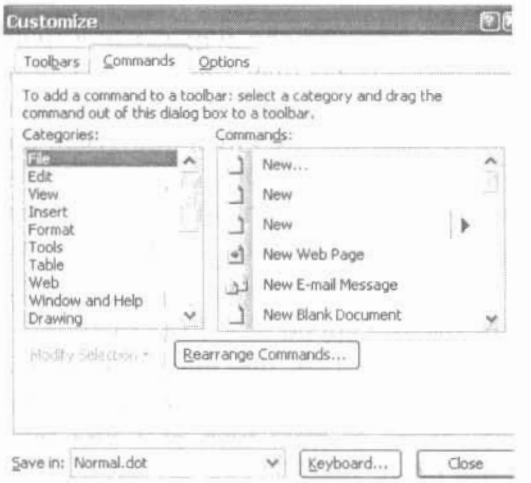
3.2. Thêm một nút mới vào thanh công cụ

Giả sử bạn luôn phải định dạng chữ, hãy thêm vào thanh Standard các nút thể hiện Font chữ hay dùng. Cách thức tiến hành như sau:

Chọn Tools - Customize (hoặc đưa chuột lên thanh công cụ rồi bấm phím phải sau đó chọn Customize) xuất hiện cửa sổ hình 1.42.

Trong cửa sổ Customize chọn chức năng Command, kéo thanh cuốn phía dưới để tìm chức năng Fonts và bấm đơn vào đó phía bên phải sẽ xuất hiện toàn bộ các font chữ có trong Word. Bấm rê một font nào đó lên thanh công cụ, tiếp đó bấm vào Modify Selection để thay đổi chữ hiện trong nút và chọn biểu tượng cho nút (xem lại phần Macro).

Khi một nút thể hiện một font chữ đã có trên thanh công cụ việc định dạng chữ sẽ trở nên rất đơn giản, chỉ cần bôi đen chữ rồi bấm vào nút font đã có.



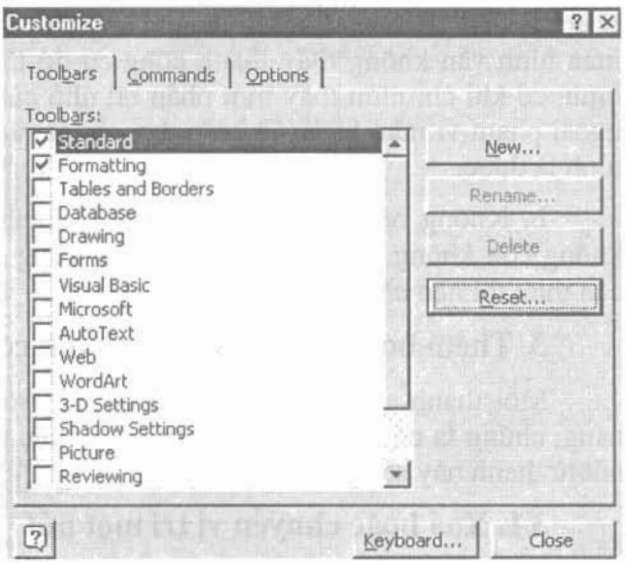
Hình 1.42

3.3. Hồi phục trạng thái ngầm định của một thanh công cụ

Đôi khi vì tò mò hay vô tình bạn phá hỏng một thanh công cụ, để khôi phục lại dạng ngầm định của thanh đó chúng ta có thể làm như sau:

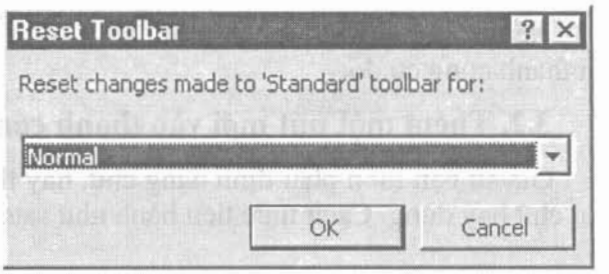
Chọn Tools - Customize (hoặc đưa chuột lên thanh công cụ rồi bấm phím phải sau đó chọn Customize) xuất hiện cửa sổ hình 1.43.

Chọn chức năng Toolbars khi đó tên các thanh công cụ xuất hiện phía dưới. Bấm đơn vào dòng chữ tên thanh công cụ để nó nằm trên nền màu xanh. (Nhớ là không bấm vào hình vuông bên trái) sau đó bấm đơn vào nút Reset phía bên phải cửa sổ, lúc này xuất hiện cửa sổ nhỏ hình 1.44.



Hình 1.43

Trong cửa sổ Reset Toolbars dưới mục chọn reset changes made to 'standard' toolbar for: chọn Normal rồi chọn OK, thanh công cụ đã chọn sẽ được khôi phục lại nguyên dạng ngầm định mà Word đã thiết kế.



Hình 1.44

4. Bảo mật văn bản

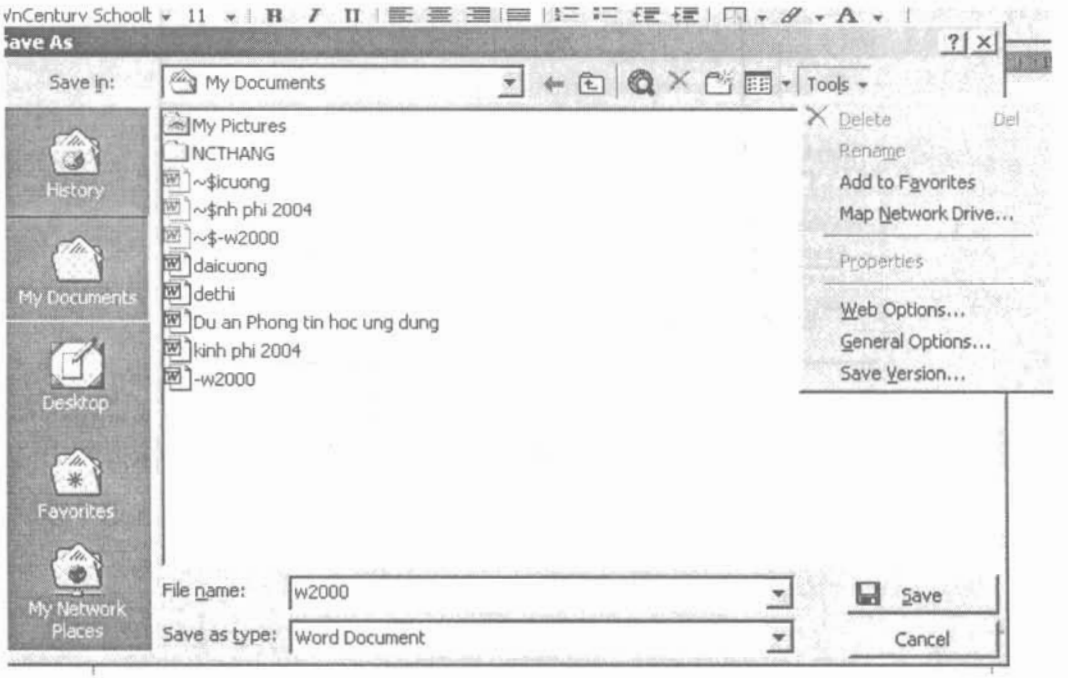
Những văn bản quan trọng không thể để cho người khác tự động mở ra cần được bảo vệ bằng mật khẩu. Bảo mật văn bản có hai cấp:

* Cho phép đọc nhưng không được sửa chữa

* Không cho phép đọc

Thao tác tiến hành:

- Chọn File – Save As xuất hiện cửa sổ hình 1.45.



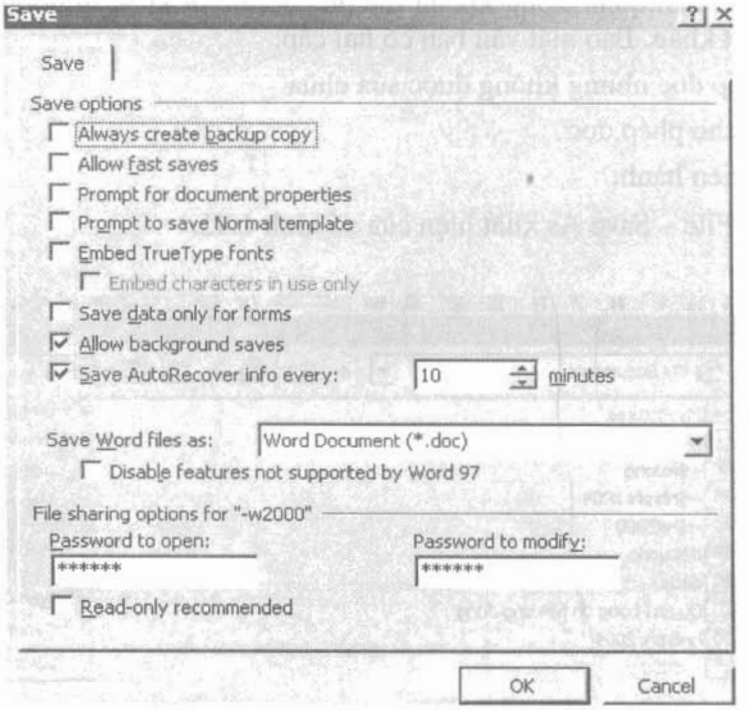
Hình 1.45

Trong cửa sổ Save As chọn Tools - chọn tiếp General Options, sẽ xuất hiện cửa sổ hình 1.46.

Mục Password to Open: Mật khẩu cho phép mở văn bản, chúng ta tuỳ chọn một mật khẩu nào đó. Nên chọn khoảng 5-6 ký tự và cần lưu mật khẩu này vào sổ để đề phòng sau một thời gian có thể bị quên. Với mật khẩu này chúng ta có thể mở văn bản ra xem nhưng không thể thêm bớt gì vào văn bản.

Mục Password to Modify: Mật khẩu cho phép sửa chữa văn bản, có thể chọn mật khẩu giống như Mục Password to Open hoặc chọn khác đi. Chọn xong bấm OK máy sẽ xuất hiện cửa sổ mới yêu cầu nhắc lại mật khẩu một lần nữa. Gõ lại đúng các mật khẩu đã chọn. Bấm OK thoát ra.

Sau khi đã chọn mật khẩu bấm Save để ghi lại văn bản kèm theo mật khẩu. Kể từ nay muốn mở văn bản ra xem chúng ta phải khai báo mật khẩu.


Save
Save
Save options
Always create backup copy
Allow fast saves
Prompt for document properties
Prompt to save Normal template
Embed TrueType fonts
Embed characters in use only
Save data only for forms
Allow background saves
Save AutoRecover info every:
10
minutes
Save Word files as:
Word Document (*.doc)
Disable features not supported by Word 97
File sharing options for "-w2000"
Password to open:
******
Password to modify:
******
Read-only recommended
OK
Cancel

Hình 1.46

Giáo trình

TIN HỌC ĐẠI CƯƠNG

Tác giả: Ts. DƯƠNG XUÂN THÀNH

Chịu trách nhiệm xuất bản: Ts. PHẠM VĂN DIỄN

Biên tập và sửa bài: Ths. NGUYỄN HUY TIẾN
NGỌC DIỆP

Trình bày bìa: XUÂN DŨNG

NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT
70 Trần Hưng Đạo – Hà Nội

In 1000 cuốn khổ 19 × 27 cm, tại Xưởng in NXB Văn hoá Dân tộc
Số đăng ký kế hoạch xuất bản: 414 – 2008/CXB/124 – 16/KHKT cấp ngày 13/5/2008
Quyết định xuất bản số: 187/QĐXB – NXBKHKT cấp ngày 8/8/2008
In xong và nộp lưu chiểu Quý III năm 2008

Tác giả Ts. Dương Xuân Thành hiện là chủ nhiệm Khoa Công nghệ Thông tin Đại học Chu Văn An.

Tốt nghiệp ngành Toán Cơ Đại học Bách khoa - Hà Nội, tác giả đã có gần bốn mươi năm kinh nghiệm giảng dạy tại một số trường Đại học, Cao đẳng trong nước.

Các giáo trình đã xuất bản của tác giả đã được sử dụng như giáo trình chuẩn cho sinh viên chuyên ngành Công nghệ Thông tin tại một số trường Đại học và Cao đẳng.

Các giáo trình đã xuất bản:

Tin học đại cương NXB Thống kê - Hà Nội, 2002

Tin học đại cương NXB Giao thông vận tải - Hà Nội, 2004

Ngôn ngữ lập trình Pascal NXB Giao thông vận tải - Hà Nội, 2004

Lập trình nâng cao NXB Nông nghiệp - Hà Nội, 2005

Hệ quản trị cơ sở dữ liệu Foxpro (Trường Đại học Nông nghiệp Hà Nội)

Cơ sở dữ liệu quan hệ (Trường Đại học Nông nghiệp Hà Nội).

0807090000085

51,000

Giá: 51.000đ